Formalizing and Studying Dialectical Explanations in Inconsistent Knowledge Bases

Abdallah Arioua

THÈSE

Pour obtenir le grade de Docteur

Délivré par l'**Université de Montpellier**

Préparée au sein de l'école doctorale **I2S**
Et de l'unité de recherche **UMR IATE et LIRMM**

Spécialité: **Informatique**

Présentée par **Abdallah Arioua**

Formalizing and Studying Dialectical Explanations in Inconsistent Knowledge Bases

Soutenue le 17/10/2016 devant le jury composé de

M. Patrice Buche	IR, HDR	INRA	Directeur
Mme. Madalina Croitoru	MCF, HDR	Univ Montpellier	Co-directrice
M. Jérôme Fortin	MCF	Univ Montpellier	Encadrant
M. Anthony Hunter	Professeur	Univ College London	Rapporteur
M. Nicolas Maudet	Professeur	Univ Pierre et Marie Curie	Rapporteur
Mme. Juliette Dibie	Professeur	AgroParisTech	Examinatrice
Mme. Leila Amgoud	DR	CNRS - IRIT	Examinatrice
M. Bernard Cuq	Professeur	Montpellier SupAgro	Invité

In the beginning, like all PhD students I set out to be a genius, but mercifully laughter intervened.

Abdallah[1]

[1] From Lawrence Durrell's original citation.

Acknowledgments

First and foremost, I thank the persons to whom I owe my existence and my success! Mother and Father, who have always been my friends and guides. I thank my whole family, my brother and my sisters, for their incredible support, I could have never achieved so much without them.

When I started my thesis I was a blank page. I learned a lot from Madalina, she was always on my side with her advices and ideas and I will never forget her comprehensiveness. I felt blessed and lucky to have her as one of my supervisors, in fact that's what everybody has told me :). Away from the thesis, I remember all those insightful discussions about interesting philosophical, personal and career matters, I really appreciate it. I thank also my two other supervisors, Patrice Buche who was always kind to me and comprehensive, he was a source of motivation to me. And Jérôme Fortin who helped me a lot by his incredible talent in formalizing ideas.

I would like to express my deepest gratitude to Anthony Hunter, Nicolas Maudet, Leila Amgoud and Juliette Dibie for making me the honor of accepting to evaluate my thesis.

I would like to thank everybody in my two research teams UMR IATE AXE 5 and GraphIK, they made me feel like home during our lunches and coffee breaks. It should be noticed that this thesis would have not been possible without the funding from the ANR DUR-DUR project and CEPIA INRA department of INRA, which allowed me to stay completely focused on my research during those three years. I thank all those with whom I collaborated in the project, It was an incredible experience to be in touch with experts from different fields, I learned a lot.

Special thanks go to my friend Nouredine Tamani, he was like a brother to me, he helped me a lot during my thesis by his ideas and advices, I don't think that I'll be able to return his favors one day. Special thanks go to my friend Hichem Debbi who inspired me with his courage and patience during his thesis, I owe him big time. I would also like to thank my friends in Algeria for their support. And, I will never regret the new friends I met here in Montpellier. Those friends are far from being forgotten, they have their places in my heart and memory. We had three years of fun, craziness and adventures, a special thanks goes to those with whom I traveled and had my best trips seeing the world, they are real men, it would have been a big loss If I had not met them.

I would like to thank all the people who helped me during my thesis and all those that I have met in the conferences, workshops and social events.

Abstract:

Knowledge bases are deductive databases where the machinery of logic is used to represent domain-specific and general-purpose knowledge over existing data. In the existential rules framework, a knowledge base is composed of two layers: the data layer which represents the factual knowledge, and the ontological layer that incorporates rules of deduction and negative constraints. The main reasoning service in such framework is answering queries over the data layer by means of the ontological layer. As in classical logic, contradictions trivialize query answering since everything follows from a contradiction (ex falso quodlibet). Recently, inconsistency-tolerant approaches have been proposed to cope with such problem in the existential rules framework. They deploy repairing strategies on the knowledge base to restore consistency and overcome the problem of trivialization. However, these approaches are sometimes unintelligible and not straightforward for the end-user as they implement complex repairing strategies. This would jeopardize the trust relation between the user and the knowledge-based system. In this thesis we answer the research question: "How do we make query answering intelligible to the end-user in presence of inconsistency?". To answer the question we consider the general framework of argumentation and we propose three types of explanations: (1) One-shot Argument-based Explanations, (2) Meta-level Dialectical Explanations, and (3) Object-level Dialectical Explanations. The first one is a set of arguments in favor or against the query in question. The two others take the form of a dialogue between the user and the reasoner about the entailment of a given query. We study these explanations in the framework of logic-based argumentation and dialectics and we study their properties and their impact on users.

Keywords: Argumentation, Inconsistency, Explanation, Dialogue Games, Existential Rules, *Datalog*±.

Résumé:

Les bases de connaissances sont des bases de données déductives où la logique est utilisée pour représenter des connaissances de domaine sur des données existantes. Dans le cadre des règles existentielles, une base de connaissances est composée de deux couches : la couche de données qui représentent les connaissances factuelle et la couche ontologique qui incorpore des règles de déduction et des contraintes négatives. L'interrogation des données à l'aide des ontologies est la fonction de raisonnement principale dans ce contexte. Comme dans la logique classique, les contradictions posent un problème à l'interrogation car d'une contradiction, on peut déduire ce que l'on veut (ex falso quodlibet).

Récemment, des approches d'interrogation tolérantes aux incohérences ont été proposées pour faire face à ce problème dans le cadre des règles existentielles. Elles déploient des stratégies dites de réparation pour restaurer la cohérence. Cependant, ces approches sont parfois inintelligibles et peu intuitives pour l'utilisateur car elles mettent souvent en œuvre des stratégies de réparation complexes. Ce manque de compréhension peut réduire l'utilisabilité de ces approches car elles réduisent la confiance entre l'utilisateur et les systèmes qui les utilisent. Par conséquent, la problématique de recherche que nous considérons est comment rendre intelligible à l'utilisateur l'interrogation tolérantes aux incohérences. Pour répondre à cette question de recherche, nous proposons d'utiliser deux formes d'explication pour faciliter la compréhension des réponses retournées par une interrogation tolérante aux incohérences. La première est dite de niveau méta et la seconde de niveau objet. Ces deux types d'explication prennent la forme d'un dialogue entre l'utilisateur et le raisonneur au sujet des déductions retournées comme réponses à une requête donnée. Nous étudions ces explications dans le double cadre de l'argumentation fondée sur la logique et de la dialectique formelle, comme nous étudions leurs propriétés et leurs impacts sur les utilisateurs en termes de compréhension des résultats.

Keywords: Argumentation, Incohérence, Explication, Jeux de Dialogues, Règles Existentielles, *Datalog*±.

Contents

List of Figures

List of Tables

1 Introduction

This thesis presents an original research in the field of Knowledge Representation and Reasoning, a central Artificial Intelligence issue. The knowledge representation we consider is the existential rules framework (*Datalog*±), a family of logical languages to represent ontologies introduced in Calì et al. (2009a,b); Chein and Mugnier (2009). This language is widely-used in the ONTOLOGY-BASED DATA ACCESS (OBDA) paradigm introduced in Poggi et al. (2008) where an ontology on top of an existing and potentially inconsistent data is used to enrich the querying process. The contribution of the thesis is the proposal of a formal model of explanation based on logic-based argumentation to explain query answers in presence of inconsistency. The model's goal is to improve the usability of knowledge-based systems that use existential rules formalism.

This chapter is structured as follows. In Section 1.1 we start by introducing the general context of the thesis and the line of development of the existential rules framework. Next in Section 1.2 we shift to a narrower context where we present how inconsistency is handled within the existential rules framework under the OBDA paradigm. Then, in Section 1.3 we present the research problem of the thesis alongside to our contributions on this regard. Finally, in Section 1.4 we conclude the chapter by highlighting the structure of the thesis.

1.1 Knowledge Representation and Reasoning

Knowledge Representation and Reasoning (KRR) is one of the main issues in Artificial Intelligence (AI). It deals with the problem of representing real-world knowledge in order to achieve human-level intelligence and reasoning faculties. The biggest dilemma in Knowledge Representation and Reasoning is the tradeoff between expressiveness and computational tractability of a given formalism (Levesque and Brachman, 1987). In fact, the difficulty of reasoning increases proportionally with the expressive power of the underlying logical language. With the rapid growth of data in the last two decades, an emergent need for tractable but yet expressive logical languages has been raised. This has particularly appeared within the intersection of Knowledge Representation and Database Systems where Deductive Databases have been firstly introduced as a logic-based formalism to improve the querying facility of classical databases with deductive functionalities (i.e. more expressiveness) (Gallaire and Nicolas, 1987; Ceri et al., 2012). It is in fact the fruit of combining logic programming with relational databases which provides a framework that is more expressive than relational databases but less expressive than logic programming systems. In a deductive database, data is represented as a set of facts written within a first-order language (Prolog-like syntax). Alongside the data, rules are added to enrich the vocabulary of the data and allow the deduction of new facts. *Datalog* is a rule-based language that is often used as an underlying expressive representation and tractable reasoning machinery for Deductive Databases (Ceri et al., 1989). According to (Ramakrishnan and Ullman, 1995, p. 2), it is the adaptation of *"...Prolog, which has a "small-data" world view of the world to a "large-data" world"*. This logical language has been recently extended to *Datalog*± to fulfill the need of the Semantic Web movement where more expressiveness is needed. *Datalog*± extends *Datalog* with (mainly) the capacity of coping with incomplete knowledge by introducing existentially quantified variables in the facts and the head of the rules, this makes it possible to refer to unknown individuals as "the person x has a mother whose name is unknown". As opposed to *Datalog*, *Datalog*± comes with another type of formulas called negative constraints which captures some sort of logical negation by forbidding certain combinations of facts. These extensions (alongside to others) make *Datalog*± general enough to capture a variety of Description Logics families (Cali et al., 2010) which are the underpinning logical formalisms for OWL/OWL2 languages. This generality promotes *Datalog*± as an adequate language to represent ontologies as shown in Calì et al. (2012). For historical accuracy, the logical interpretation of Conceptual Graphs of Sowa (1976) in Chein and Mugnier (2009) has yielded the same formalism as *Datalog*± but under the name of **existential rules framework**. In the thesis we may use the two names interchangeably.

Nowadays, the existential rules framework is widely-used in the paradigm of Ontology-based Data Access (OBDA) where it provides satisfactory results with respect to the reasoning issue on how to query data while taking ontological knowledge. However, under this paradigm the problem of data **inconsistency** emerges and manifests itself as logical contradictions, namely in form of constraints violations in the factual part. Since OBDA is mainly used in data exchange and data integration, inconsistency is more likely to occur in the data part rather than the ontological part (rules and constraints) since the ontology is assumed to be well-written by domain experts. On top of that, the size of the ontology is rather small when compared to the size of the data.

The problem of inconsistency has challenged the KRR formalisms. New approaches to tolerate and cope with inconsistency became a must. In the next section we give a brief introduction on the main approaches.

1.2 Inconsistency Tolerance

Inconsistency management is a well-established research discipline in KRR that motivated and challenged the KRR community for decades. In fact, it dates since the pre-Socratic Greek era where the concept of logical contradictions appeared in the sayings of Parmenides of Elea as reported by Plato:

> *The great Parmenides from beginning to end testified... "Never shall this be proved - that things that are not are".* (Plato, Sophist, 237A)

The problem is in fact due to the principle of explosion in logic (ex falso quodlibet) which states that "from contradiction, anything follows". Consequently, given an inconsistent knowledge base $\mathcal{K}$ in a logical language $\mathcal{L}$ then one can deduce every sentence from the knowledge base according to Classical Logics, more precisely, If one assumes that ψ and $\neg\psi$ are both true then one can say that any formula ϕ is true. This can be proven as follows: $\psi \vee \phi$ is true because ψ is true (by assumption). But $\neg\psi$ is also true by assumption, therefore, for $\psi \vee \phi$ to be true ϕ must be true, hence ϕ is true.

In KRR several approaches to handle inconsistency have been proposed in the literature. The two main approaches are Coherence-based approaches and Dung-style Logic-based Argumentation. Please notice that this domain of research is vivid and full of approaches. In this introduction we limit ourselves to the above-mentioned approaches as the main problem of the thesis is not inconsistency management.[1]

[1] For more details on the subject we refer the reader to Martinez et al. (2013); Bertossi et al. (2005).

In the Coherence-based approaches (firstly introduced in Rescher and Manor (1970)), maximal consistent subsets (MCSs) of the knowledge base are constructed and new non-classical consequence relations are defined to infer from the knowledge base. A formula is a *universal* consequence of the knowledge base if and only if it is the logical consequence of all MCSs. A formula is a *credulous* consequence of the knowledge base if and only if it is the consequence of at least one MCS. Different maximality criteria (set-inclusion or cardinality) give rise to different types of non-classical consequence relations. In the same line of research, other approaches such as Benferhat et al. (1993, 1995) appeal to a primal concept of argumentation in order to define new consequence relations that are mainly based on the idea of MCSs but differ in their productivity (they may validate more or less formula compared to the universal consequence relation).

Dung-style Logic-based Argumentation approaches (also called Coalition approaches in Bertossi et al. (2005)) have been widely used to reason in presence of inconsistency. An argumentation framework à la Dung (1995) is defined as a set of arguments and an attack relation among them. The logic-based version of Dung's framework regards an argument as a tuple of a hypothesis and conclusion built from a given knowledge base $\mathcal{K}$ over a base logic. The approach proceeds by computing from $\mathcal{K}$ the set of all arguments. Next, it computes the attack relation among them (which is grounded on inconsistency), then it produces coalitions of arguments called extensions (i.e. sets of non-conflicting arguments that defend one another). From the extensions, inconsistency-tolerance is defined as follows: a formula is entailed from $\mathcal{K}$ if and only if there exists an argument that belongs to all extensions and whose conclusion entails the formula in question. In Cayrol (1995) it has been shown that by picking a specific attack relation (attack on the hypothesis, i.e. *undercut*) Dung-style Logic-based Argumentation approaches generalize the Coherence-based approaches of Rescher and Manor (1970). The work in Amgoud and Besnard (2013); Vesic (2013) generalizes this result for any logic-based argumentation framework that is grounded on a Tarskian logic.

Despite their equivalence, the Coherence-Based approaches of Rescher and Manor (1970) have gained an enormous interest in the field of Database Systems where the concept of *Maximal Consistent Subsets* took the name of *Data Repairs* and the *Universal Consequence* took the name of Consistent Query Answering semantics (CQA) (Arenas et al., 1999; Calvanese et al., 2005; Bertossi, 2006; Chomicki, 2007; Bertossi, 2011).[2] The reason that the

[2]For an exhaustive treatment of the subject we refer the reader to Bertossi (2011).

Coherence-based approaches have gained interest is due to their constructiveness. Put differently, they are not in rupture with Classical Logics as much as the other approaches such as Paraconsistent Logics (Priest, 1979), Default Logics (Reiter, 1980), etc. In fact constructing maximal consistent subsets over an inconsistent knowledge base is an attempt to restore consistency by eliminating inconsistencies as less as possible. The reason to do so is to be able to reuse Classical Logics machinery in reasoning as the universal consequence relation, or any other coherence-based consequence relations, will be defined in terms of the classical consequence relation.

Later on, Data Repairs and the Consistent Query Answering semantic were adapted to suit the OBDA setting in Lembo et al. (2010) where Description Logics is used as a representation formalism for the ontology. On the same line of research, Lukasiewicz et al. (2012) handled the case of OBDA where the ontology is represented within *Datalog*±.

To clarify how CQA works, let us give an informal example. Please notice that the example is for illustration purposes to introduce the intuition behind the approach. Formal examples will be introduced in future chapters.

Example 1.2.1. *Consider the following inconsistent knowledge base $\mathcal{K}$ about a hotel. Following the existential rules framework it is composed of a set of facts, set of rules and set of negative constraints. Imagine that we have only these information and there are no means by which we can verify their truthfulness and reliability. And imagine that we are interested in knowing whether there was a person in Room 1408 or not at 8pm.*
We have the following set of facts:

(F_1): The light was On in Room 1408 at 8pm.

(F_2): John has seen Alice in Room 1480 at 8pm.

(F_3): John was in vacation the whole month.

(F_4): There was no electricity in Room 1408 at 8pm.

(F_5): Video footages at 8pm have shown Alice in Room 1408.

And the following set of rules:

(R_1): If the light was On in room x at a given time y then there was a person in room x at time y.

(R_2): If there was no electricity in room x at a given time y then there was no person in room x at time y.

(R_3): If video footages show a person at a given time y in room x then there was a person in room x at time y.

(R_4): If someone has seen another person in room x at a given time y then this person was in room x at time y.

And the following set of constraints:

(N_1): It is impossible that the light was On in room x and there was no electricity in room x.

(N_2): It is impossible for a person x to see someone in a place z at a given time y given that x was not in the same place as z at time x.

Inconsistency is syntactically defined as the violation of one of the constraints.[3] One can see that $\mathcal{K}$ is inconsistent because F_1 and F_4 violate together the constraint N_1 and F_2 and F_3 violate together N_2. Now the user may be interested in querying the knowledge base $\mathcal{K}$ using the queries:

- "$\mathcal{Q}_1$: was there a person in Room 1408 at 8pm?"
- "$\mathcal{Q}_2$: was there no person in Room 1408 at 8pm?"

The problem is that under the classical query answering in OBDA we would get a *yes* answer to $\mathcal{Q}_1$ because F_1 and R_1 allows the deduction of such answer. However, we get a *yes* answer too for $\mathcal{Q}_2$ using F_4 and R_2. As one can see, this is a clear contradiction.

To cope with such problem, the CQA semantics constructs data repairs on the set of facts. In OBDA it is assumed that the facts are less reliable than the rules and constraints whose reliability is taken for granted. It is the reason why data repairs are built only over the facts.

Repair $\mathcal{P}_1 = \{F_1, F_2, F_5\}$:

The light was On in Room 1408 at 8pm. John has seen Alice in Room 1480 at 8pm. Video footages at 8pm have shown Alice in Room 1408.

Repair $\mathcal{P}_2 = \{F_1, F_3, F_5\}$:

The light was On in Room 1408 at 8pm. John was in vacation the whole month. Video footages at 8pm have shown Alice in Room 1408.

[3] This is equivalent to the non-existence of a model for $\mathcal{K}$ in the model-theoretic interpretation.

Repair $\mathcal{P}_3 = \{F_4, F_3, F_5\}$:

There was no electricity in Room 1408 at 8pm. John was in vacation the whole month. Video footages at 8pm have shown Alice in Room 1408.

Repair $\mathcal{P}_4 = \{F_4, F_2, F_3\}$:

There was no electricity in Room 1408 at 8pm. John has seen Alice in Room 1480 at 8pm. Video footages at 8pm have shown Alice in Room 1408.

To answer the queries CQA uses the following reasoning strategy:

- If a query $\mathcal{Q}$ has a *yes* answer over all repairs then the answer to the query $\mathcal{Q}$ under the CQA semantics is *yes* ($\mathcal{Q}$ is accepted under CQA).
- If a query $\mathcal{Q}$ has at least one *no* answer over all repairs then the answer to the query $\mathcal{Q}$ under the CQA semantics is *no* ($\mathcal{Q}$ is not accepted under CQA).

Following CQA, the query $\mathcal{Q}_1$ has a yes answer whereas the query $\mathcal{Q}_2$ has a no answer.

It seems that with such semantics, inconsistency is dealt with to a certain degree. However, another problem not as much as addressed as inconsistency emerges. It is when the user asks "why $\mathcal{Q}_1$ has a yes answer" or "why $\mathcal{Q}_2$ has a no answer?" under CQA. The user here is asking for an explanation about the answers of some queries under the inconsistency-tolerant semantics CQA. The next section is devoted to present this problem in details.

1.3 Research Problem

The problem of inconsistency is not the only problem that faces Ontology-based Data Access. Another important issue is to explain query answering under inconsistency-tolerant semantics. This is motivated by the fact that reasoning using these approaches in the context of OBDA is a complex procedure that deploys different repairing strategies that may not seem intuitive for the user when querying the knowledge base. Indeed the user may think that a query should have a specific answer while the inconsistency-tolerant semantics cuts off this answer as it is deemed inconsistent. For instance, the user may have thought that there was no person in Room 1408, so he/she wants to know why the answer was "yes there was a person in Room

1408 at 8pm". In case of absence of explanations, the usability of OBDA is jeopardized and the trust in the inconsistency-tolerant semantics is at stake (McGuinness and Patel-Schneider, 1998).

We focus on the notion of explanation in the context of one particular approach of the coherence-based approaches in OBDA, namely the CQA semantics. We restrict ourselves to the CQA semantics for the following reasons:

- CQA semantics is a widely-used, well-developed and thoroughly-investigated semantics that lies in the intersection of many disciplines. Most notably: KRR, Database Systems and Semantic Web.
- All coherence-based approaches in the OBDA setting are variants of CQA. In fact the other approaches are attempts to reduce its computational intractability:
  - Intersection semantics (Bienvenu, 2012; Lembo et al., 2010).
  - Lazy semantics (Lukasiewicz et al., 2012).
  - k-defeater and k-supporter (Bienvenu and Rosati, 2013).
  - Preferred and Cardinality-Based semantics (Bienvenu et al., 2014).
  - Non-objection (Bouraoui et al., 2016).

  This fact has been confirmed in the unifying framework of Baget et al. (2016) which incorporates all the approaches in one framework.
- It has a tight relation with other formalisms such as Default Logics (Arioua et al., 2015b) and Logic-Based Argumentation in existential rules (Croitoru and Vesic, 2013).

So the research problem we consider in this thesis is:

> Research problem
> *How do we make Consistent Query Answering intelligible to the end-user?*

The answer to this research problem is:

> Solution
> *We do so by providing explanations to facilitate the understanding of Consistent Query Answering.*

The claim of the thesis is:

> Claim
> *Formal Argumentation can serve as a solution to this problem. Moreover, it provides an added explanatory value with respect to the state-of-the-art approaches.*

One may be tempted to tackle such research problem by adapting classical methods of explanation in KRR, especially from Description Logics (DLs). In what follows we give the state-of-the-art approaches and we show why they are not suitable to solve this research problem.

Approaches in McGuinness and Borgida (1995); Borgida et al. (2000); Schlobach et al. (2003) are close to our research problem. Precisely, they tackle the problem of explaining "why the knowledge base is inconsistent?" as opposed to "why the query $\mathcal{Q}$ is (not) accepted?". The solutions proposed for their problem is to compute the so-called Minimal Inconsistent Subsets which are those sets of formulas that are responsible for the inconsistency. To explain why the knowledge base of Example 1.2.1 is inconsistent, these solutions would give the following explanation (S_1 and S_2 counts as one explanation):

> S_1: *The light was On in Room 1408 at 8pm. There was no electricity in Room 1408 at 8pm.*
>
> S_2: *John was in vacation the whole month. John has seen Alice in Room 1480 at 8pm.*

To show that this solution is not suitable for our problem consider the query:

- "$\mathcal{Q}_3$: was John in vacation the whole month?"

The answer to this query is "no"under the CQA semantics. The explanation above would fail to answer the question "why the query $\mathcal{Q}_3$ has a no answer?" because S_1 is completely irrelevant as an explanation of the query $\mathcal{Q}_3$. In addition, the explanation lacks elaboration which is an important aspect when interacting with users. It is important to say that we are not implying the futility of these approaches, but rather we are delineating their area of impact.

Other works in DLs such as Baader et al. (2007); Schlobach (2005) focused on explaining why a concept is subsumed (implied) by another concept using *Axiom Pinpointing* and *Concept Pinpointing*. This is done by computing minimal subsets of formulas (i.e. concepts and axioms) from the knowledge base that entails the formula in question. This approach has been adapted for existential rules in Arioua et al. (2014a) where an explanation is defined as a deduction path that starts from a minimal set of facts and rules and ends with the query to be explained. This approach suffers from the following shortcoming:

- **Soundness:** consider the query $\mathcal{Q}_1$ which has a yes answer under CQA. If we look for a minimal subset of the knowledge base that allows the entailment of $\mathcal{Q}_1$ one could point out directly the set of formulas $\{F_1, R_1\}$. However, this is not true because the real reason behind is in fact the minimal subset $\{F_5, R_3\}$. Put differently, under the CQA semantics the query should be entailed from all repairs and F_1 is not in all repairs, hence $\{F_1, R_1\}$ is not a correct explanation. On the contrary, F_5 is in all repairs therefore $\{F_5, R_3\}$ is the *actual* explanation.

The first contribution of the thesis aims at overcoming this shortcoming.

1.3.1 Contribution 1: One-shot Argument-based Explanations

As we have stated in Section 1.2, Coherence-based approaches and Dung-style Logic-based Argumentation are equivalent. Recently, this result has been confirmed by Croitoru and Vesic (2013) for CQA semantics and Dung-style Logic-based Argumentation under existential rules within OBDA. This equivalence relation is very interesting as argumentation is well-known for its explanatory power as stated by Modgil and Caminada (2009a).

In Dung's abstract model of argumentation (Dung, 1995), arguments acquire a justification state (according to a semantics). An argument is skeptically accepted if it belongs to all extensions. It is credulously accepted if it belongs to at least one extension. In the logical instantiation of this model, the arguments are usually seen as tuples of logical formulas called hypothesis and conclusion. In this case another type of justification state is introduced. We say that a formula is skeptically accepted if it is the conclusion of a skeptically accepted argument and it is credulously accepted if it is the conclusion of a credulously accepted argument. In Amgoud et al. (2008) a new acceptance called universal has been introduced that states that a conclusion is universally accepted if it is the conclusion of different arguments that are distributed over all extensions. This universal acceptance in the existential rules framework is shown by Croitoru and Vesic (2013) to be exactly the same as CQA semantics for queries in OBDA as a consequence of a deeper equivalence result that states that a bijection exists between extensions and repairs. However, there were no exact characterizations of why a query could be universally accepted and not skeptically accepted, or vice-versa. This in fact appears to be very important to introduce and define formally the concept of explanations.

In Chapter 3 we follow Croitoru and Vesic (2013) and investigate the formal properties of their instantiation. We show that it enjoys interesting properties that are very important for the other contributions of this thesis. We characterize universal acceptance in terms of what we call *blocks* and *proponent sets*:

- **Proponent set:** a proponent set is a minimal set of arguments in favor of the query that contains an argument from each extension. A query is universally accepted if and only if it has a proponent set (Arioua and Croitoru, 2016c; Arioua et al., 2015a).
- **Block:** a block is an admissible set of arguments that attacks all the supporters of the query. A query is not universally accepted if and only if it has a block (Arioua and Croitoru, 2016c; Arioua et al., 2015a).

Since finding blocks or proponent sets are necessary and sufficient reasons to determine universal acceptance, consequently CQA entailment, we use them as **explanations** to show why a query has a yes or no answer under CQA. It worth noticing that this approach has been investigated separately at the same time by Bienvenu et al. (2016) using another framework. The approach relies on the concept of a *cause* which is a minimal set of facts that entails the query. Then CQA explanations are defined as set of causes with certain properties. It is clear that causes here are the counterpart of arguments in our framework and explanations are equivalent to blocks and proponent sets. From now on we may refer to our solution to mean the two.

As we discovered during the thesis, this solution, as beneficial as it seems to be, could be improved to overcome the following issues:

1. **Computational burden:** computing all extensions is inefficient and computationally hard. This makes computing explanations inherently hard as it takes the extensions as an input.
2. **Lack of interactiveness:** these explanations are one-shot. They are presented to the user as a set of arguments with no further elaboration. In fact, they cannot handle *user's expectation failure*. In this condition the user may have some expectations about the query, or some prior

knowledge that may made him/her think that the query should have a yes (resp. no) answer under CQA. Therefore, he/she maybe interested in asking follow-ups. In other words, he/she may be interested in an explanation that takes the form of a dialogue, i.e. a **dialectical explanation**.[4]

The solution to these two issues should:

- Allow the computation of blocks on the fly instead of computing the extensions first.
- Provide an added explanatory value with respect to interactiveness as opposed to One-shot Argument-based Explanations.
- Be engaging to the point that it allows the user to be involved in the process of discovering the answer of the query.

In the next section we describe the second contribution of the thesis regarding these issues.

1.3.2 Contribution 2: Meta-level Dialectical Explanations

Dialectical proof theories in argumentation (Modgil and Caminada, 2009b) are common ways to prove the justification state of an argument without passing through the process of computing extensions. Dialectical proofs are dialogues between two fictitious players, one is called `PRO` and tries to establish the justification state of an argument (skeptical state for instance) and the other is called `OPP` and tries to refute such state. Dialectical proofs provide a procedural description of how the argument would acquire its justification state (i.e. they are explanations). And since they take the form of dialogues between two players, the user can be engaged in such dialogue by taking the role of one of the players. However, to the best of our knowledge there is no dialectical proof theory for universal acceptance in logic-based argumentation in general and logic-based argumentation within the extensional rules framework.

In Chapter 4 we propose a dialectical proof theory for universal acceptance and we prove its completeness and soundness. The significance of this contribution is twofold:

[4] DIALECTICS: the art of investigating or discussing the truth of opinions. DIALECTICAL: relating to the logical discussion of ideas and opinions. The Oxford English Dictionary. `http://www.oxforddictionaries.com/definition/english/dialectic`. (08/07/2016).

- It handles an unsolved problem in Dung-style Logic-based Argumentation frameworks under the existential rules framework.
- It handles the issues that face one-shot argument-based explanations.

These two claims shed light on our experimental hypothesis with respect to this contribution:

> *Meta-level dialectical explanations enjoy an added explanatory value when compared to one-shot argument-based (a.k.a cause-based) explanations with respect to Consistent Query Answering Explanation.*

To validate this hypothesis we present in Chapter 4 an experimental evaluation on the impact of one-shot argument-based explanation and meta-level dialectical explanation on users. The result is significant and promising as a validation.

Meta-level dialectical explanations are referred to as *meta-level* for a reason. Their goal is to get the user to understand why the query has a yes or no answer under CQA. It is more justificatory in the sense that it tries to convince or persuade the user that the answer under the CQA semantics is justified. Therefore, the object of the explanation is in fact the semantics itself not the content of the query. Consequently, Meta-level Dialectical Explanations are content-independent. This makes them not suitable for "educative" explanations. Those explanations that answer the questions "What is a video footage?" or "Why the sky is blue?", etc. And even if they can answer "why" questions they are unable to provide the user with appropriate locutions within the dialogue such as declaring whether he/she understands or not the explanation. Moreover, it does not give the reasoner the possibility to track the state of understanding of the user to stay coherent and precise.

In the next section we present the last contribution of the thesis on this regard.

1.3.3 Contribution 3: Object-level Dialectical Explanations

In this contribution (Chapter 5) we introduce Object-level Dialectical Explanations which complement meta-level dialectical explanations on the educative part. The motivation of this contribution stems from an application problem.

The DUR-DUR project aims at improving the French Durum Wheat quality while reducing pesticides and fertilizers. The project is multidisciplinary with different partners from different disciplines. Our task was to integrate knowledge that comes from different parties in one knowledge base about Durum Wheat then make it available for the parties for querying about Durum Wheat related knowledge. As the project is multidisciplinary, those who work, for instance on pasta transformation, may find difficulties in understanding Durum Wheat related knowledge as it lies out of their expertise. Another problem was the difficulty to acquire knowledge from experts as they are not expert in existential rules formalism and find it hard to explore a knowledge base written in a logical language. Therefore, extending meta-level dialectical explanations to the object-level is indeed needed, in particular for the following reasons:

- It would facilitate the communication between different experts in different disciplines.
- It would improve knowledge acquisition as the content of the knowledge base would be better exposed when the reasoner explains in a goal-directed and rule-governed way. This would consequently help in reducing inconsistencies.

In this contribution we propose a dialogue model of explanation that is based on Walton's dialogue model of explanation; Walton (2004, 2007, 2011, 2016). We define its syntax and semantics and we instantiates it on the existential rules framework.

The significance of our work lies within the following points:

- It shows how argumentation theory can go beyond other approaches in handling usability issues in knowledge-based systems.
- It attempts to solve a practical problem and shows the significance of Formal Dialectics as an important discipline in formal argumentation.

We present in Chapter 5 a use case with Agronomy experts on the utility of using object-level dialectical explanations in knowledge acquisition that shows promising results for their utility.

1.4 Thesis Structure

In addition to this chapter which provides the context of our research, the thesis contains seven chapters.

- Chapter 2 introduces necessary preliminaries on Dung's abstract model of argumentation proposed in his seminal paper Dung (1995). It explains the different semantics proposed by Dung, namely, *admissible, complete, stable, preferred and grounded* and their coincidence. We present the relations established in the literature among these classes. Exploring such relation is very important to investigate the gain in terms of computational complexity. It explains how an argument can be regarded as skeptically, credulously accepted, or rejected under a given semantics.

- Chapter 3 presents the study of Dung-style logic-based argumentation in the existential rules framework. First, we start by introducing the logical language of existential rules. Then, we proceed by presenting the instantiation of Dung's abstract model on this logical language by Croitoru and Vesic (2013). We prove that this instantiation enjoys interesting properties, i.e. finiteness, coherence, relative groundedness and non-triviality. The chapter also presents the first contribution of the thesis where *One-shot Argument-based Explanations* are introduced as a precise characterization of the universal acceptance. It further investigates the satisfaction of the recent postulates proposed in Amgoud (2014) and recalls the representation theorem established by Croitoru and Vesic (2013); Vesic (2013) between preferred/stable extensions and data repairs of Lembo et al. (2010); Bienvenu (2012). This chapter builds upon works published in Arioua and Croitoru (2016c); Arioua et al. (2015a).

- Chapter 4 is about the second contribution of the thesis where we propose Meta-level Dialectical Explanations by solving the problem of universal and non-universal acceptance in logic-based argumentation framework through a dialectical proof theory (dialogue game). We prove the soundness and completeness of the dialectical proof theory and we study the dispute complexity of dialectical proofs alongside other interesting properties. We empirically evaluate the effect of meta-level dialectical explanations on users with respect to different criteria. We report how they impact the accuracy of users when faced with inconsistent situations. Moreover, we investigate how the users evaluate meta-level dialectical explanations with respect to clarity and intelligibility. This chapter builds upon the work published in Arioua and Croitoru (2016c) and parts from Arioua and Croitoru (2016b); Arioua et al. (2014c,b).

- Chapter 5 is about *Object-level Dialectical Explanations*. As opposed to Meta-level Dialectical Explanations, Object-level Dialectical Explanations are domain-specific explanations which are meant to use the domain knowledge stored in the knowledge base to answer the user's requests for explanations. It presents how the state-of-the-art explanatory dialogue model fails to capture certain desirable aspects. For that matter, we propose a dialogue model for explanations called `EDS`. We formally define its protocol's syntax and semantics and we investigate the role of commitments in such dialogue and their relation to termination and success. We discuss how this dialogue model can be extended with argumentative faculties so that it can account for user's objections against explanations. We present a use case within the DUR-DUR project that shows how object-level dialectical explanations can be used in knowledge acquisition and inconsistency resolution. This chapter builds upon the work published in Arioua and Croitoru (2015).

- Chapter 6 presents another contribution with respect to the DUR-DUR project which consists of the construction of a domain-specific knowledge base about Durum Wheat under the existential rules framework. It presents the implementation of a prototype called DALEK (DiALectical Explanations in Knowledge bases) that implements One-shot Argument-based Explanation, Object-level Dialectical Explanations and certain aspects of Meta-level Dialectical Explanations. This chapter describe the work published in Arioua and Croitoru (2016a).

- Chapter 7 concludes the thesis and presents a number of interesting future research problems.

2
Abstract Argumentation

In this chapter we provide an introduction to Dung's abstract model of argumentation proposed in his seminal paper Dung (1995) through an introductory example. Next, in Section 2.2 we recall the formal definitions of the semantics proposed by Dung, namely, *admissible, complete, stable, preferred and grounded*. It turns out that these semantics may coincide. Their coincidence gives birth to different argumentation classes. In Section 2.4 we present the relations between these classes. Exploring such relation is very important to investigate the gain in terms of complexity.

Different semantics give different interpretations of argument acceptability and justification state. In Section 2.3 we conclude the chapter by recalling how an argument can be regarded as skeptically, credulously accepted, or rejected.

2.1 Introduction

Humans are always concerned with debating and arguing as it constitutes an important part of our day-to-day communications. Alongside to other types of dialogues, argumentation dialogues are of great impact on our lives. It is within which opinions are confronted against each other and arguments are advanced to support them. It turns out that this complex process of arguing comes down, as Dung has concluded, to a very simple principle: "The one who has the last word laughs best". To better illustrate the point consider the following example from Dung (1995), where two persons I and A whose countries are at war about who is responsible for blocking negotiation in their region.[1]

I: My government cannot negotiate with your government because your government doesn't even recognize my government.

[1]The author does not side with any political view, this is only an illustrative example.

A: Your government doesn't recognize my government either.

The person **I** here tries to give an argument that supports the point that the one who blocks the negotiation is **A**'s government because it doesn't recognize **I**'s government. **A** in its turn uses the same argument to counter attack **I**'s argument using the same *way of reasoning*. At this stage, neither **I** nor **A** can claim victory in the debate. Consider the following continuation:

I: But your government is a terrorist government.

This argument presents another attack on **I**'s argument. If the exchange stops here, then **I** clearly has the "last word", which means that he has successfully argued that **A**'s government is responsible for blocking the negotiation (Dung, 1995, pp. 322).

The goal of Dung's work was to give a scientific account of this basic principle: "The one who has the last word laughs best", and to explore possible ways for implementing this principle on computers.

2.2 Abstract Argumentation Frameworks

The model of Dung is an abstract model where the structure of arguments and the type of attack is not defined. It takes as input a set of arguments and a pre-constructed binary relation that represents attack between arguments.

Definition 2.2.1 (Argumentation framework)**.** *An argumentation framework is a pair $\mathcal{H} = (\mathcal{A}, \mathcal{X})$ where $\mathcal{A}$ is a set of arguments and $\mathcal{X}$ is a binary relation over $\mathcal{A}$. Given two arguments $a, b \in \mathcal{A}$ we denote by $a\mathcal{X}b$ or $(a, b) \in \mathcal{X}$ that the argument a attacks b.*

Remark 2.2.1. *If $\mathcal{A}$ is finite then $\mathcal{H}$ is called a finite argumentation framework. $\mathcal{H}$ is called finitary if each argument $a \in \mathcal{A}$ is attacked by a finite set of arguments. Note that in finitary argumentation frameworks the set of arguments could be infinite.*

An argumentation framework can be seen as a directed graph where vertices represent arguments and edges represent attack between arguments.

Example 2.2.1 (Dung (1995))**.** *The exchange between **I** and **A** can be represented by an argumentation framework $\mathcal{H} = (\mathcal{A}, \mathcal{X})$ as follows: $\mathcal{A} = \{i_1, i_2, a\}$ and $\mathcal{X} = \{(i_1, a), (a, i_1), (i_2, a)\}$ with i_1 and i_2 denoting the first and the second argument of **I**, respectively, and a_1 denoting the argument of **A**. The figure 2.1 represents the argumentation framework (a.k.a argument graph).*

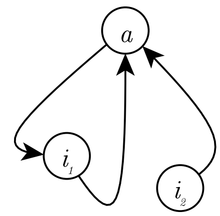


Figure 2.1: The corresponding argument graph of Example 2.2.1.

Definition 2.2.2. *We say a set of arguments S attacks an argument b if there exists an argument $a \in S$ such that $(a, b) \in \mathcal{X}$. If there is an argument $c \in S$ such that $(b, c) \in \mathcal{X}$ and S attacks b we say that S* ***defends*** *c. We also the use following notations:*

- $\texttt{range}^+(a) = \{b \mid (a, b) \in \mathcal{X}\}$.
- $\texttt{range}^-(a) = \{b \mid (b, a) \in \mathcal{X}\}$.
- $\texttt{range}^+(S) = \bigcup_{a \in S} \texttt{range}^+(a)$.
- $\texttt{range}^-(S) = \bigcup_{a \in S} \texttt{range}^-(a)$.

Example 2.2.2 (Cont'd)**.** $\{i_1, i_2\}$ *attacks a.*

A rational agent accepts only arguments for which she has a rational reason to do so. This reason is called *acceptability condition*, which stipulates that a rational agent accepts only arguments which she can defend from all possible attacks. This condition is defined with respect to the concept of *argumentation semantics*. The latter refers to a set of criteria applied on a set of arguments. Two different methods are proposed in the literature to define semantics, extension-based of Dung (1995) and labeling-based of Caminada (2006). The latter is based on labeling the arguments with specific labels, namely, **in**, **out**, **und** meaning that the argument is accepted, rejected and undecided respectively. In this thesis we focus on the extension-based approaches which is more declarative. It defines explicitly what an acceptable argument means under some specific criteria. Examples of these semantics, the admissible, complete, grounded, preferred and stable semantics due to Dung (1995). Other semantics such as prudent, recursive, semi-stable and ideal (among others) can be found in Baroni and Giacomin (2009). We limit the scope of the thesis to those semantics which are defined in Dung (1995).

Any extension-based semantics is based on the principle of conflict-freeness which translates the idea that for a set of arguments to be considered as an extension it should be able to stand together. That means there is no attack between the elements of the same extension.

Definition 2.2.3 (Conflict-freeness)**.** *We say a set of arguments S is conflict-free if and only if there are no $a, b \in S$ such that $(a, b) \in \mathcal{X}$.*

Example 2.2.3 (Cont'd)**.** $\{i_1, i_2\}$ *is conflict-free.*

A set of non-conflicting arguments can be seen as an agent's position. In fact, it does not correspond exactly to a position, since a position should stand on its own Baroni and Giacomin (2009). This means that a set of non-conflicting arguments S should counterattack any outside attack by means of arguments only from S. This corresponds to the notion of acceptability and admissibility in Dung (1995).

Definition 2.2.4 (Admissibility)**.** *Given an argumentation framework* $\mathcal{H} = (\mathcal{A}, \mathcal{X})$.

- *An argument $a \in \mathcal{A}$ is **acceptable** with respect to a set of arguments $S \subseteq \mathcal{A}$ if and only if $\forall b \in \mathcal{A}$; if $(b, a) \in \mathcal{X}$ then S attacks b.*
- *A conflict-free set of arguments S is **admissible** if and only if every argument $a \in S$ is acceptable with respect to S.*

Acceptability is defined with respect to the defense criteria. Admissibility is based on acceptability and conflict-freeness, therefore an admissible set of arguments is a set of non-conflicting arguments that defends all its elements. That means that all arguments of S are acceptable with respect to S. Such set is called an *admissible extension.*

Example 2.2.4. *In Example 2.2.1 we have 3 admissible extensions: $\emptyset$, $\{i_1\}$,$\{i_2\}$, $\{i_1, i_2\}$. But $\{a\}$ is not an admissible extension since it does not defend itself from i_2.*

Note that every argumentation framework has at least one admissible set, the empty set.

Definition 2.2.5 (Complete semantics)**.** *Given an argumentation framework $\mathcal{H} = (\mathcal{A}, \mathcal{X})$. An admissible set of arguments $S \subseteq \mathcal{A}$ is a complete extension if and only if $\forall a \in A$ if S defends a then $a \in S$.*

Example 2.2.5 (Cont'd)**.** *The set $\{i_1, i_2\}$ is a complete extension. $\{i_1\}$ and $\{i_2\}$ are not.*

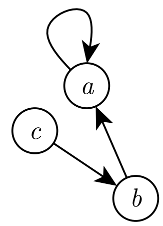


Figure 2.2: Empty stable extensions.

A more refinement of the admissible and complete semantics is the stable semantics where *aggressiveness* is imposed.

Definition 2.2.6 (Stable semantics)**.** *Given an argumentation framework* $\mathcal{H} = (\mathcal{A}, \mathcal{X})$*. A set of arguments* $S \subseteq \mathcal{A}$ *is a stable extension if and only if* S *is conflict-free and for all* $a \in \mathcal{A}$ *such that* $a \notin S$*,* S *attacks* a*.*

This means S attacks all arguments outside of S. One can observe that in Example 2.2.5 where the set $\{i_1, i_2\}$ is a stable extension. Because of the aggressive behavior an argumentation framework may have no stable extension.

Example 2.2.6. *Consider the argumentation framework* $\mathcal{H} = (\mathcal{A}, \mathcal{X})$ *of Figure 2.2 where* $\mathcal{A} = \{a, b, c\}$ *and* $\mathcal{X} = \{(c, b), (b, a), (a, a)\}$*. It is clear that* $\mathcal{H}$ *has no stable extension, however the sets* $\emptyset$ *and* $\{c\}$ *are admissible and complete.*

To avoid the problem of having no extension under the stable semantics (i.e. extension emptiness), the aggressiveness criterion is relaxed in the preferred semantics where a preferred extension is a maximal admissible set.

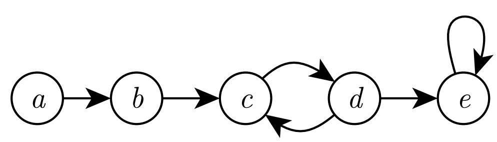
Definition 2.2.7 (Preferred semantics)**.** *Given an argumentation framework* $\mathcal{H} = (\mathcal{A}, \mathcal{X})$*. A set of arguments* $S \subseteq \mathcal{A}$ *is a preferred extension if and only if* S *is a maximal (w.r.t set-inclusion) admissible extension of* $\mathcal{H}$*.*

Example 2.2.7. $\{c\}$ *in the previous example is a preferred extension. In Example 2.2.1 the set* $\{i_1, i_2\}$ *is a preferred extension which is the superset of the admissible extensions* $\{i_1\}$ *and* $\{i_2\}$*.*

An argument may be accepted with respect to an extension and rejected with respect to another. Dung has proposed another semantics which is called the grounded semantics. This semantics is a unique-extension semantics, meaning that an argumentation framework has only one grounded extension and an argument can be either accepted or rejected by this unique extension.

Definition 2.2.8 (Grounded semantics). *The grounded extension of an argumentation framework $\mathcal{H}$ is the least (w.r.t set-inclusion) complete extension.*

Example 2.2.8. *Consider the following argumentation framework.*



The admissible extensions are $\{d\}, \{a\}, \{a, d\}, \{a, c\}$. The complete extensions are $\{a\}, \{a, c\}$ and $\{a, d\}$. The preferred extensions are $\{\}, \{a, c\}$ and $\{a, d\}$. Observe that the least complete extension of the complete extensions is $\{a\}$ which is the grounded extension. And finally we have one stable extension $\{a, d\}$.

One can see that there is a strong relation between the semantics. For instance, in the previous example all complete extensions are admissible extensions but not vice versa, and all preferred extensions are complete but not vice versa. In what follows we show the relation between the aforementioned semantics.

Theorem 2.2.2 (Dung (1995)). Given an argumentation framework $\mathcal{H} = (\mathcal{A}, \mathcal{X})$. The following hold:

- Every stable extension is preferred.
- Every preferred extension is complete.
- Every grounded extension is complete.
- Every complete extension is admissible.

We conclude the following properties about the grounded, admissible and preferred semantics.

Theorem 2.2.3 (Dung (1995)). Let $\mathcal{H} = (\mathcal{A}, \mathcal{X})$ be an argumentation framework. The following hold:

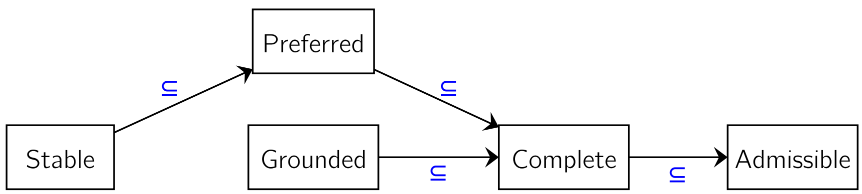


Figure 2.3: The inclusion relation between the semantics.

- $\mathcal{H}$ has at least one preferred extension.
- $\mathcal{H}$ has a unique grounded extension (which may be empty).
- The grounded extension of $\mathcal{H}$ is a subset of any preferred extension of $\mathcal{H}$.
- For each admissible set S of $\mathcal{H}$, there exists a preferred extension $\mathcal{E}$ of $\mathcal{H}$ such that $S \subseteq \mathcal{E}$.

Table 2.1 summaries the semantics and their essential criteria. The criteria are as follows: CF means the extensions are conflict-free, DF means they defend all their elements, INCDF means they include what they defend, MAX means they are maximal w.r.t $\subseteq$, AGR refers to *aggressiveness* and it means they attack all what is outside and ADM refers to admissibility. Please note that since the table is only for illustration purposes the criteria are not completely dependent, some of them are derivable from others (for instance, ADM from CF and DF).

2.3 Justification State

Argumentation frameworks take as an input a set of arguments and an attack relation, they give as an output a set of extensions under a specific semantics. Taking into account this output, arguments can be classified with respect to their membership in the resulting extensions. Therefore, we distinguish the following three justification state.

Definition 2.3.1 (Justification state)**.** *Given a semantics x and argumentation framework $\mathcal{H}$. Let $\texttt{Ext}_x(\mathcal{H})$ denote the set of all extensions of $\mathcal{H}$ under the semantics x. An argument $a \in \mathcal{A}$ is:*

	CF	DF	ADM	INCDF	MAX	AGR
Admissible	×	×	×			
Complete	×	×	×	×		
Stable	×	×	×	×		×
Preferred	×	×	×	×	×	
Grounded	×	×	×	×		

Table 2.1: Semantics with respect to criteria. × means the criterion is satisfied.

- *Skeptically accepted if and only if* $a \in \bigcap \mathcal{E}_i$*, such that* $\mathcal{E}_i \in \mathtt{Ext}_x(\mathcal{H})$*.*
- *Credulously accepted if and only if* $a \in \bigcup \mathcal{E}_i$*, such that* $\mathcal{E}_i \in \mathtt{Ext}_x(\mathcal{H})$*.*
- *Rejected if and only if it is not credulously accepted.*

A skeptically accepted argument is an argument that is accepted from all possible standpoints (extensions), whereas a credulously accepted argument that may be accepted in some extensions and rejected in others. A rejected argument is referred to as an *overruled* argument. It is clear that every skeptically accepted argument is also credulously accepted but not vice versa.

Example 2.3.1 (Cont'd Example 2.2.8)**.** *Under the preferred semantics, the argument* a *is skeptically accepted and the arguments* d *and* c *are credulously accepted. The argument* b *is rejected.*

Note that the justification state can change with respect to the semantics under consideration.

2.4 Coincidence between Semantics

In general, reasoning with abstract argumentation frameworks is hard (Dunne and Wooldridge (2009); Dimopoulos et al. (1999)). However, this task becomes easier when certain properties are verified. Namely, the preferred and stable semantics coincide (*coherence*), the coincidence between intersection of preferred extensions and the grounded extension (*relative groundedness*),

Decision problem	Complexity
Is S admissible?	P
Is S a stable extension?	P
Is S a preferred extension?	CO-NP-C
Does $\mathcal{H}$ have any stable extension?	NP-C
Is x in some preferred extension?	NP-C
Is x in some stable extension?	NP-C
Is x in every stable extension?	CO-NP-C
Is x in every preferred extension?	Π_2^p-C
Is x in the grounded extension?	P
Is $\mathcal{H}$ coherent?	Π_2^p-C

Table 2.2: Computational complexity of certain decision problems. $\mathcal{H}$ is an argumentation framework, x is an argument and S a set of arguments. Results taken from Dunne and Bench-Capon (2002); Dunne and Wooldridge (2009).

coincidence between complete, grounded, preferred and stable with a unique extension (*well-foundedness*) and the existence of one empty preferred extension (*triviality*). In this section we recall each property and show the main sufficient conditions for such coincidence proved in Dung (1995); Dunne and Bench-Capon (2002); Doutre (2002).

Let us start with the coherence property of Dung (1995). One significant benefit of coherence has been shown in Vreeswik and Prakken (2000) where a proof mechanism for establishing skeptical acceptance has been provided for coherent argumentation frameworks. Therefore, satisfying such property in certain argumentation frameworks comes with that benefit. In addition, as it is know from Dunne and Bench-Capon (2002) the computational complexity of the skeptical membership problem is co-NP-complete for stable and Π_2^p-complete for preferred. In this case, an enormous reduction in complexity is gained where we jump from the the second level of the polynomial hierarchy Π_2^p-complete to the first level Π_1^p-complete (i.e. co-NP-complete). To plot a full picture about the computational benefit, Table 2.2 shows the computational problems with their computational complexities.

Definition 2.4.1 (Coherence)**.** *An argumentation framework $\mathcal{H}$ is said to be*

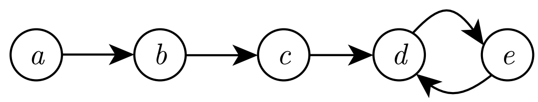


Figure 2.4: Coherent argumentation framework.

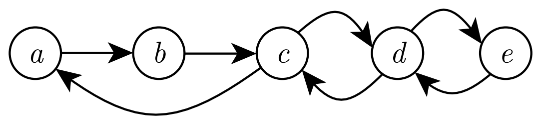


Figure 2.5: Non-coherent argumentation framework.

coherent if and only if all its preferred extensions are stable extensions. We denote by COHERENT *the class of argumentation frameworks that are coherent.*

Example 2.4.1. *Figure 2.5 presents an argumentation framework $\mathcal{H}$ that is not coherent, observe that $\mathtt{Ext}_p(\mathcal{H}) = \{\{a,d\},\{e\}\}$ which is not equal to $\mathtt{Ext}_s(\mathcal{H}) = \{\{a,d\}\}$. The argumentation framework $\mathcal{H}'$ of 2.4 is coherent where $\mathtt{Ext}_p(\mathcal{H}') = \mathtt{Ext}_s(\mathcal{H}') = \{\{a,c,e\}\}$.*

The problem is how one can tell from looking at the graph whether a given argumentation framework is coherent? Unfortunately, according to Dunne and Bench-Capon (2002) there is no tractable procedure (unless P=NP) that can preform such check. However, certain classes of argumentation frameworks for which there are efficiently testable properties that suffice to guarantee coherence are identified in the literature.

Dung identified the classes *uncontroversial* and *limited controversial* argumentation frameworks for which coherence is verified. These two classes are based on two simple concepts called indirect defense/attack.

Definition 2.4.2 (Indirect defense/attack)**.**

- *We say an argument b* ***indirectly attacks*** *a if there exists a finite sequence $(a_0, \ldots, a_{2n+1})$ such that (1) $a_0 = a$ and $b = a_{2n+1}$ and (2) for each i, $0 \leq i \leq 2n$, a_{i+1} attacks a_i.*

- *We say an argument b* ***indirectly defends*** *a if there exists a finite sequence $(a_0, \ldots, a_{2n})$ such that (1) $a_0 = a$ and $b = a_{2n}$ and (2) for each i, $0 \leq i \leq 2n$, a_{i+1} attacks a_i.*

An argument is controversial if it defends and attacks (indirectly) in the same argument.

Definition 2.4.3 (Controversy). *An argument b is said to be controversial with respect to a if b indirectly attacks a and indirectly defends a. An argument is controversial if it is controversial w.r.t at least one argument.*

Example 2.4.2. *In the argumentation framework of Figure 2.5 the argument c is controversial because it indirectly attacks/defends e. To see why, we give the following sequences:*

- (e, d, c) *where as one can see c attacks d and d attacks e (c indirectly defends e).*
- (e, d, c, b, a, c) *(c indirectly attacks e).*

Uncontroversy and limited controversy is defined as follows.

Definition 2.4.4 (Uncontroversial and limited controversial).

1. *An argumentation framework is uncontroversial if none of its arguments is controversial.*
2. *An argumentation framework is limited controversial if there exists no infinite sequence of arguments $a_0, \ldots, a_n, \ldots$ such that a_{i+1} is controversial with respect to a_i.*

We denote by UNCONTROVERSIAL *and* LCONTROVERSIAL *the class of argumentation frameworks that are uncontroversial and limited controversial respectively.*

Example 2.4.3. *In the argumentation framework of Figure 2.4 one can see that there is no controversial argument.*

As stated by Dung (1995), every uncontroversial argumentation framework is limited controversial but not vice versa. And the two of them are coherent. Put clearly:

Theorem 2.4.1 (Dung (1995)). UNCONTROVERSIAL $\subset$ LCONTROVERSIAL $\subseteq$ COHERENT.

In Dunne and Bench-Capon (2002) a larger class that subsumes Dung's controversial argumentation frameworks has been identified. In what follows we introduce this class.

We denote by NODDCYCLE the class of argumentation frameworks that have no simple directed cycle of odd length. Recall that in graph theory a cycle is a path whose source node is identical to the goal node. The length of a cycle is equal to the number of edges in that cycle. A simple cycle is a cycle where repetitions of vertices and edges are not allowed.

Theorem 2.4.2 (Dunne and Bench-Capon (2002)). NODDCYCLE $\subseteq$ COHERENT.

Example 2.4.4. *Observe that Figure 2.4 has no cycle of odd length.*

Doutre (2002) has shown that NODDCYCLE comprises limited controversial argumentation frameworks. Put differently, a limited controversial argumentation framework has no cycle of odd length.

Theorem 2.4.3 (Doutre (2002)). LCONTROVERSIAL $\subseteq$ NODDCYCLE.

Another interesting class of argumentation frameworks that has been proven to be coherent by Coste-Marquis et al. (2005) is symmetric argumentation frameworks (denoted as SYMMETRIC). Symmetric frameworks are those argumentation framework whose attack relation is symmetric.

Theorem 2.4.4 (Coste-Marquis et al. (2005)). SYMMETRIC $\subseteq$ COHERENT.

Another class of coherent argumentation frameworks is the one where the complete, preferred, stable and grounded coincide resulting a unique non-empty extension. The corresponding class of such case is called well-founded argumentation frameworks and denoted as WFOUNDED. It is first identified by Dung (1995).

Definition 2.4.5 (Well-foundedness). *An argumentation framework $\mathcal{H} = (\mathcal{A}, \mathcal{X})$ is well-founded iff there is no sequence $a_0, \ldots, a_n, \ldots$ such that for each i, $(a_{i+1}, a_i) \in \mathcal{X}$.*

For finite argumentation frameworks this class corresponds exactly to the argumentation frameworks with no cycles (Doutre (2002)).

Theorem 2.4.5 (Dung (1995)). Every well-founded argumentation framework has exactly one complete extension which is grounded, preferred and stable.

The preferred can coincide with another semantics. In what follows we look at the coincidence between the intersection of preferred extensions and the grounded extension (Dung (1995)).

Definition 2.4.6 (Relative groundedness). *We say that an argumentation framework $\mathcal{H}$ is relatively grounded if its grounded extension coincides with the intersection of all preferred extensions. We denote by* RGROUNDED *the class of argumentation frameworks that are relatively grounded.*

Class	Name	Description
COHERENT	Coherent	$\forall\mathcal{H}, \texttt{Ext}_p(\mathcal{H}) = \texttt{Ext}_s(\mathcal{H})$.
NODDCYCLE	No simple odd-length cycle	$\forall\mathcal{H}$, $\mathcal{X}$ has no simple odd-length cycle.
UNCONTROVERSIAL	Uncontroversial	$\forall\mathcal{H}$, $\mathcal{H}$ has no controversial arguments.
LCONTROVERSIAL	Limited controversial	$\forall\mathcal{H}$, there is no sequence $a_0, \ldots, a_n, \ldots$ s.t for each i, a_{i+1} is controversial w.r.t a_i.
RGROUNDED	Relatively grounded	$\forall\mathcal{H}, \texttt{Ext}_g(\mathcal{H}) = \bigcap \texttt{Ext}_p(\mathcal{H})$.
WFOUNDED	Well-founded	$\forall\mathcal{H}$, there is no sequence $a_0, \ldots, a_n, \ldots$ s.t $(a_{i+1}, a_i) \in \mathcal{X}$ for each i.
SYMMETRIC	Symmetric	$\forall\mathcal{H}$, $\mathcal{X}_\mathcal{H}$ is symmetric and irreflexive.

Table 2.3: Classes of argumentation frameworks studied in the literature. Note that these classes are with respect to those argumentation frameworks that contain at least one argument. p and s refers to preferred and stable respectively.

This is an interesting property since in this case the problem of "Is x in every preferred extension?" reduces down to checking weather "Is x in the grounded extension?" which is polynomial.

The following theorem is provided by Dung (1995) that stipulates that every uncontroversial argumentation framework is relatively grounded.

Theorem 2.4.6 (Dung (1995)). UNCONTROVERSIAL $\subseteq$ RGROUNDED.

Example 2.4.5. *The argumentation framework of Example 2.2.8 is relatively grounded because $\mathcal{E} = \{a\}$ coincides with the intersection of the preferred extensions, i.e. $\{a, d\} \cap \{a, c\}$.*

When the argumentation framework has one preferred extension which is the empty extension then it is called trivial. Note that these kind of argumentation frameworks are not coherent because the empty extension is not a stable extension. Therefore one can use the different testable properties of coherence to check triviality.

Table 2.3 summaries the known classes of argumentation frameworks in literature. Figure 2.6 shows the relation between the known classes of coherent frameworks. Figure 2.7 shows the relation between major classes, i.e. coherent, relatively grounded and unique preferred extension. Please note that this figure is taken from Doutre (2002).

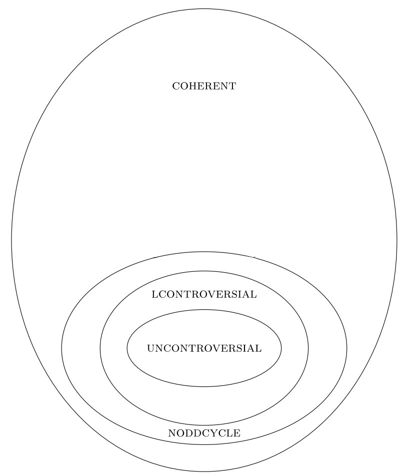


Figure 2.6: Inclusion between classes of coherent argumentation frameworks.

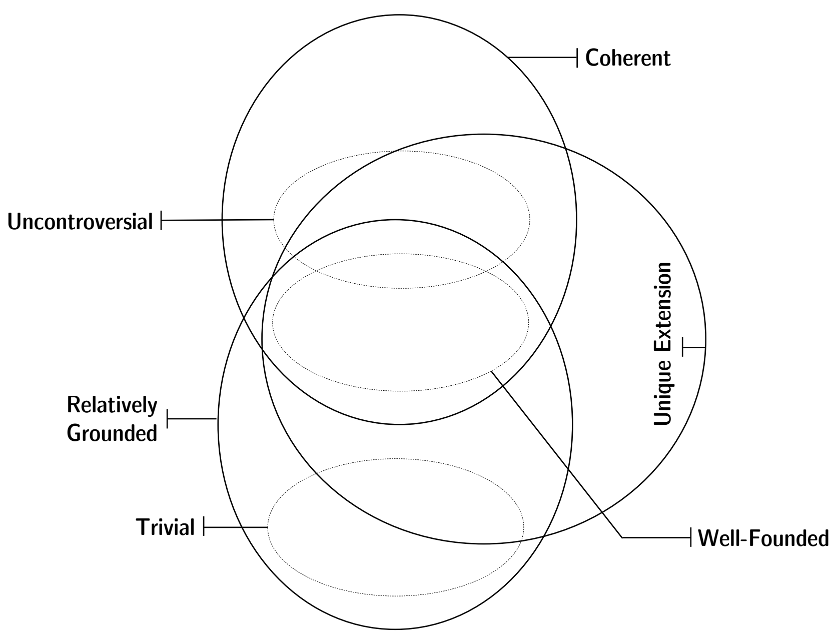


Figure 2.7: Relation between major classes of argumentation frameworks.

2.5 Conclusion

In this chapter we have presented Dung's abstract argumentation framework. In this frameworks arguments are abstract entities related with an abstract concept of attack relation. We have recalled the different semantics proposed in Dung (1995) and we have explored the relation established in the literature between them. We also presented different properties that an argumentation framework may verify, namely, finiteness, controversy, coherence, relative groundedness, well-foundedness and triviality.

An interesting relation between the preferred and the stable semantics is when they coincide, the corresponding family of argumentation frameworks that verifies this property are called coherent argumentation frameworks. We have seen some sub-families of such family. Finally, we have recalled the main stream justification state which are studied in the literature, i.e. skeptical, credulous and rejected.

As already emphasized, Dung's model is abstract and the structure arguments and the type of the attack relation is left unspecified. In the next chapter we study the instantiation of Dung's model on a specific logical language called the existential rules framework (Baget et al., 2011b; Chein and Mugnier, 2009; Calì et al., 2012), we study its properties and its output.

3 Logic-based Argumentation

In this chapter we study Dung-style logic-based argumentation in the logical language of existential rules (*Datalog*±). We precisely characterize its output to introduce a formal definition of *One-shot Argument-based Explanations*. We show how this particular instantiation enjoys the properties studied in the previous chapter, i.e. finiteness, coherence, relative groundedness and non-triviality. After that, we further investigate the satisfaction of the recent postulates proposed in Amgoud (2014). Finally, we recall the representation theorem established by Croitoru and Vesic (2013); Vesic (2013) between preferred/stable extensions and data repairs of Lembo et al. (2010); Bienvenu (2012) which is crucial to exploit the explanatory power of argumentation in explaining query acceptance under the Consistent Query Answering semantics.

3.1 Introduction

In Dung's abstract frameworks, arguments are regarded as abstract entities. There is no specified structure for arguments. Also, there is no semantics specification of what is an attack between arguments. They are considered to be given as an input. Generally, we are interested in constructing an instantiated argumentation framework where we start by building arguments from a knowledge base under a given logical language using a given logic, then the different interactions between the arguments are identified according to the type of attack we are willing to consider. Four major logical approaches have been studied in the literature. Assumption-based argumentation frameworks (ABA) (Bondarenko et al., 1993), ASPIC/ASPIC+ (Modgil and Prakken, 2013), DeLP (García and Simari, 2004) and the Deductive Argumentation (Besnard and Hunter, 2008). The first three approaches are rule-based approaches where arguments are constructed from a knowledge base with defeasibile and strict rules. The fourth approach is more oriented towards Classical Logics where an argument is perceived as a tuple (H, c) of set of premises and a conclusion. In this approach two major classes are proposed in the literature. Those approaches that define logic-based argumentation over an abstract logic (Tarskian) like Amgoud and Besnard (2010), and those approaches that define argumentation over a concrete logic like; Propositional Logics in Amgoud and Cayrol (1998) or First-Order Logics in Besnard and Hunter (2008, 2014). In this thesis we are interested in the logic-based approach. More precisely, the second one. We place ourselves in a concrete setting called Ontology-based Data Access (OBDA, (Poggi et al., 2008)). In this setting the ontology is used to "access" different data sources. These sources are solely consistent but mutually inconsistent. It is taken for granted in this setting that the ontology is consistent. In the next section we introduce the logical language that accounts for such setting. Next, in Section 3.3 we deal with the instantiation of Dung's abstract argumentation frameworks over this setting.

3.2 Logical Language: Existential Rules Framework

There are two major approaches to represent an ontology in the OBDA setting. The first one is Description Logics such as $\mathcal{EL}$ (Baader et al., 2005) and DL-Lite, families (Calvanese et al., 2007). The second is rule-based languages such as *Datalog*± (Calì et al., 2012), a generalization of *Datalog* (Ceri et al., 1989) that allows for existentially quantified variables in rule's head. Despite *Datalog*± undecidability when answering conjunctive queries, different decidable fragments are studied in the literature (see Baget et al. (2011a)). These fragments generalize the aforementioned Description Logics families and overcome their limitations by allowing any predicate arity as well as cyclic structures. Here we follow the second method for its expressiveness. The main goal of this section is to introduce such logical language, which will serve as a *base* logic for the abstract argumentation framework.

The guidelines are as follows, in Subsection 3.2.1 we introduce the syntactical building blocks of the language and we show how querying facilities are available in such language. Next, to increase the expressiveness we account in Subsection 3.2.2 for the ontological part where rules and negative constraints are taken into account. However, as the OBDA setting is highly prone to inconsistencies we present in Subsection 3.2.3 how inconsistency is dealt with in the literature.

3.2.1 The language, facts and queries

We consider *the positive existential* syntactic fragment of first-order logic $\texttt{FOL}(\exists, \wedge)$ (Chein and Mugnier (2009); Baget et al. (2011b)). Its language $\mathcal{L}$ is composed of some formulas built with the usual quantifiers $(\exists, \forall)$ and *only* the connectors, implication $(\rightarrow)$ and conjunction $(\wedge)$. A special-purpose constant that denotes the falsity $\perp$ is used. There is no disjunction and negation.

We consider usual first-order vocabularies with constants but no other function symbols as follows.

Definition 3.2.1 (Vocabulary). *Consider a vocabulary composed of three disjoint sets* $\boldsymbol{Voc} = (\mathcal{C}, \mathcal{P}, \mathcal{V})$*, where* $\mathcal{C}$ *is a finite set of constants,* $\mathcal{P}$ *is a finite set of predicates and* $\mathcal{V}$ *is an infinite set of variables. A function* $\boldsymbol{ar} : \mathcal{P} \rightarrow \mathbb{N}$ *associates a natural number* $\boldsymbol{ar}(p)$ *with each predicate* $p \in \mathcal{P}$ *that defines the arity of* p*.*

- *A term* t *over* $\boldsymbol{Voc}$ *is a constant* $t \in \mathcal{C}$ *or a variable* $t \in \mathcal{V}$*.*
- *An atomic formula (or atom) over* $\boldsymbol{Voc}$ *is of the form* $p(t_1, ..., t_n)$ *where* $p \in \mathcal{P}$*,* $\boldsymbol{ar}(p) = n$ *and* $t_1, ..., t_n$ *are terms.*
- *A ground atom is an atom with no variables.*
- *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).*

We denote by $\vec{x}$ *a sequence of variables* $(x_1, \ldots, x_n)$*.*

Since we are in the setting of knowledge bases, constant symbols with different names represent different individuals (unique name assumption). This is safe to assume since our data often come from a relational database where constants that represent individuals are meant to be unique. It is to be noted that this logical language is negation-free. We use uppercase letters for constants and lowercase letters for variables.

Example 3.2.1 (Atoms and conjuncts). *Consider the following vocabulary* $\mathcal{C} = \{John\}$*,* $\mathcal{P} = \{student, teacher, teaches\}$ *and an infinite set of variables* $\mathcal{V} = \{x_1, x_2, x_3, \ldots\}$*. Then,* $teaches(John, x_1)$ *is an atom,* $teacher(John)$ *is a ground atom,* $teaches(John, x_1) \wedge teacher(John)$ *is a conjunct and* $teacher(John) \wedge teaches(John, Tom) \wedge student(Tom)$ *is a ground conjunct.*

One way to represent knowledge about the world is to grasp factual knowledge. This kind of knowledge is deemed the most basic form of knowledge, for instance the information *"John teaches Tom"* is basic or atomic. In logic programming and deductive databases this is called a *fact*, since it represents a basic form of knowledge no wonder that classically *ground atom* are used to represent facts. To account for incomplete knowledge such as "John teaches a student that we don't know the name of", a fact is extended so that it may contain existentially quantified variables and not only constants (e.g. Baget et al. (2011b)).

Definition 3.2.2 (Fact)**.** *A fact on* ***Voc*** *is the existential closure of a conjunction of atoms over* ***Voc****.*

So "John teaches a student that we don't know the name of" is represented as:

$$\exists x_1(teacher(John) \wedge student(x_1) \wedge teaches(John, x_1))$$

where x_1 is an existentially quantified variable. Note that we may omit quantifiers in facts as there is no ambiguity (they are all existentially quantified). So the existential variables permit to represent unknown values which is an interesting property in this language. In addition, in the context of OBDA and Semantic Web we cannot assume that we can name all individuals.

Notation 3.2.1. Let F be a fact, 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 $F = \exists x \exists y(r(x) \wedge p(A, y) \wedge r(x))$ can be seen as $\{p(A, y), r(x)\}$ where $vars(F) = \{x, y\}$ and $terms(F) = \{x, y, A\}$. From now on we may use the set notation and the logical notation interchangeably to denote a fact.

Arbitrary sets of *ground* facts $\mathcal{F}$ are in fact relational databases that store factual knowledge about a given domain. The reason to store knowledge is to be able answer queries about different aspects of such domain. In what follows we recall the notions of *substitution* and *homomorphism* between facts. Then we show how these two notions are used to evaluate queries over a given set of facts $\mathcal{F}$.

Definition 3.2.3 (Substitution and homomorphism)**.** *Given a set of variables $\mathcal{V}$ and a set of terms $\mathcal{T}$, a* ***substitution*** *σ of $\mathcal{V}$ by $\mathcal{T}$ (notation $\sigma : \mathcal{V} \rightarrow \mathcal{T}$) is a mapping from $\mathcal{V}$ to $\mathcal{T}$. Given a fact F, $\sigma(F)$ denotes the fact* *obtained from F by replacing each occurrence of $x \in \mathcal{V} \cap vars(F)$ 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 $\sigma(F) \subseteq F'$.*

Let us take an example to better clarify this.

Example 3.2.2 (Homomorphism)**.** *Consider the following vocabulary* $\boldsymbol{Voc} = (\mathcal{C}, \mathcal{P}, \mathcal{V})$*:*

- $\mathcal{C} = \{A, B\}$.
- $\mathcal{P} = \{q, r\}$.
- $\mathcal{V} = \{x, y, z, \ldots\}$ *is infinite set of variables.*

We have the following set of facts over the vocabulary $\boldsymbol{Voc}$*:*

- $F = \{q(A, x)\}$ *where* $terms(F) = \{A, x\}$ *and* $vars(F) = \{x\}$.
- $F' = \{q(A, B), r(A)\}$ *where* $terms(F') = \{A, B\}$ *and* $vars(F') = \emptyset$.

Consider $vars(F)$ and $terms(F')$ as our set of variables and set of terms. We have two possible substitutions.

- $\sigma_1 = \{(x, A)\}$.
- $\sigma_2 = \{(x, B)\}$.

Where x is substituted by A in σ_1 and by B in σ_2. Let us see which of these substitutions is a homomorphism from F to F':

- *When we apply σ_1 on F we get $\sigma_1(F) = \{q(A, A)\}$.*
- *When we apply resp. σ_2 on F we get $\sigma_2(F) = \{q(A, B)\}$.*

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 $\sigma_2(F) = \{q(A, B)\}$.

As in database systems, we can query our initial set of facts $\mathcal{F}$ using *queries*. *Conjunctive queries* are the basic and more frequent queries. Let us recall the definition of Baget et al. (2011b).

Definition 3.2.4 (Queries). *A conjunctive query (CQ) has the following form: $\mathcal{Q} = \texttt{ans}(x_1, \ldots, x_k) \leftarrow B$, where B (the "body" of $\mathcal{Q}$) is a fact, and $x_1, \ldots, x_k$ occur in $vars(B)$ and* `ans` *is a special k-ary predicate, whose arguments are used to build an answer. Given a set of facts $\mathcal{F}$, an answer to $\mathcal{Q}$ in $\mathcal{F}$ is a tuple of constants $(A_1, \ldots, A_k)$ such that there is a homomorphism σ from B to $\mathcal{F}$, with $(\sigma(x_1), \ldots, \sigma(x_k)) = (A_1, \ldots, A_k)$. If $k = 0$, i.e. $\mathcal{Q} = \texttt{ans}() \leftarrow B$, $\mathcal{Q}$ is called a Boolean conjunctive query, the unique answer to $\mathcal{Q}$ is the empty tuple if there is a homomorphism from B to $\mathcal{F}$, otherwise there is no answer to $\mathcal{Q}$.*

Without loss of generality, we restrict our work to BCQs as they are polynomially equivalent to CQs. Consequently the term "query" from now on refers to BCQ unless stated otherwise. Please note that a BCQ $\mathcal{Q}$ can be shortly referred to by its body B. So instead of writing $\mathcal{Q} = \texttt{ans}() \leftarrow student(John)$ we may write $\mathcal{Q} = student(John)$.

A complementary way to represent knowledge is rules. We would like to enrich our set of facts with a set of rules that encode certain domain-specific knowledge. These rules are regarded as an ontological layer that reinforces the expressiveness of the knowledge base by encoding the so-called intentional knowledge in databases. In what follows we see the added value of rules (existential ones) over other formalisms then we show how we can perform deduction (query answering) in presence of existential rules.

3.2.2 Adding rules and negative constraints

Rules have been extensively used in knowledge-based and expert systems. Rules are logical formulas which allow us to infer facts from other facts. An example of a rule is: "If x is a cat then x is an animal". In order to be general, rules often contain variables. To be even more general, rules should account for unknown individuals. For instance " If x is a cat then x has a mother y and a father z". These are called existential rules (Baget et al., 2011b; Calì et al., 2012) and the ability to represent unknown individuals is also known in database community as *value invention* (Abiteboul et al., 1995). This in fact captures the case where some information are incomplete and some individuals are unknown. In the above example, we still know that x has a mother but we fail to know who she is as the variable y is an existential one.

Overall, what we need to do when we have a set of facts supplied by a set of existential rules and negative constraints is to use these rules to deduce all possible knowledge while respecting the negative constraints. The output of such procedure is a set of facts that extends the first one. Therefore, query answering can be easily done over this new set of facts using homomorphisms.

In what follows we define the logical form of *existential rules* and *negative constraints.* Then we explain how do we use rules to derive new facts using *rule application.* And finally we explain the *saturation* procedure that performs a breadth-first application of all rules on the set of facts.

Definition 3.2.5 (Existential rules)**.** *Recall that we denote by $\vec{x}$ a sequence of variables. An* existential rule *(or simply a rule) is a closed formula of the form $R = \forall \vec{y}((\forall \vec{x} B) \rightarrow (\exists \vec{z} H))$, where B and H are conjuncts with $vars(B) = \vec{x} \cup \vec{y}$, and $vars(H) = \vec{z} \cup \vec{y}$.*

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. Given a rule R, *body*(R) and *head*(R) should not be empty.

Rules exhibit different syntactical structures. According to this syntactical structure their expressiveness varies. The well-known Horn clauses of *Datalog* (Ceri et al., 1989) and Prolog restrict the head of the rule to one atom. Existential rules overcome such limitation by allowing more than one atom in the head of the rule alongside to the possibility of representing existential variables. It is to be noted that this form of rules is also known as tuple-generating dependencies in database community; Fagin (2009).

When it comes to other formalisms such as Description Logics, existential rules are more expressive as they can represent complex relations between individuals.[1] Consider the following existential rule:

$$R = \forall y_1 y_2 \ (siblingOf(y_1, y_2) \rightarrow \exists z_1 parentOf(z_1, y_1) \wedge parentOf(z_1, y_2))$$

This cannot be expressed in Description Logics because of the "cycle on variables"; Chein and Mugnier (2009)), i.e. despite the possibility to say y_1 is linked to z_1 by $parentOf$ and y_2 is linked to z_1 by $parentOf$ there are no means in DL by which we can say that they are related to the same z_1.

Another important aspect of the existential rules framework is the possibility of having *unrestricted predicate arity.* This is very important because it can help us in adding contextual information such as provenance, trust, etc. and also it facilitates a direct translation of database relations.

[1] For an extensive study on the relation between different families of DLs and existential rules please see Mugnier and Thomazo (2014); Calì et al. (2012).

It comes as no surprise that this expressiveness is at the expense of decidability. In fact, entailment is undecidable for general existential rules (Beeri and Vardi, 1981). However, many classes of existential rules that ensure decidability (while keeping expressiveness) have been studied (see (Baget et al., 2011b)). In this thesis and for practical reasons we work on such classes.

To represent knowledge about the world one should account for *negative knowledge*, i.e. knowledge that dictates how things are not ought to be. The existential rules is negation-free which makes it hard to represent such kind of knowledge. In database systems the notion of integrity constraints is used to forbid certain invalid inputs and to preserve the semantics of the data. In the existential rules we find a logical account of integrity constraints under the name of negative constraints (Calì et al., 2012).

Definition 3.2.6 (Negative constraint)**.** *A negative constraint (or simply a constraint) is a rule of the form* $N = \forall \vec{x}(B \rightarrow \bot)$.[2]

Negative constraints are very important as they serve as logical devices to detect inconsistencies in the factual part of the knowledge base. In fact and as we will see in Subsection 3.2.3, the firing of a negative constraint is interpreted as a presence of inconsistency in the knowledge base.

Compared to Description Logics, negative constraints in the existential rules framework fully captures *concept disjointness* of DLs. In fact, negative constraints are more expressive than concept disjointness. Consider the case where an individual a can belong to at most two concepts but not three at the same time. So we may forbid that a belongs to A, B and C together but nevertheless it can belong to A and B or B and C or A and C. This form of constraints cannot be expressed in DL-Lite for instance. However, in more expressive DL families (such as $\mathcal{EL}$) this can be simulated by means of other higher concepts (e.g. we can consider "A and B" as a new concept called AB). The problem with such *tweak* is that it necessitates the modification of the initial knowledge base which makes it hard for the user to understand the newly introduced concepts.

Example 3.2.3. *An example of a negative constraint:*

$$N = retiredFrom(x, y) \wedge worksIn(x, y) \rightarrow \bot$$

It is impossible that a person is retired from an establishment and she is still working in this establishment.

[2] We may sometimes omit quantifiers in the rules and constraints and write $R = B \rightarrow H$ and $N = B \rightarrow \bot$.

Up to now, we have seen three different forms of knowledge, facts, rules and negative constraints. Let us now define a knowledge base.

Definition 3.2.7 (Knowledge base). *A knowledge base over a vocabulary **Voc** is a tuple $\mathcal{K} = (\mathcal{F}, \mathcal{R}, \mathcal{N})$ of finite sets of facts, rules and negative constraints respectively.*

In order to be able to perform deduction, rules must be used jointly with facts to produce new facts.

Example 3.2.4. *Consider the following rule and fact:*

$$R_1 : teaches(x, y) \wedge student(y) \rightarrow teacher(x)$$

$$F_1 : teaches(John, Tom) \wedge student(Tom)$$

From these two formulas we produce a new fact $teacher(John)$.

In logic, this is a pure application of the well-know Modus ponens inference rule, in *Datalog* it is referred to as the Elementary Production Principle (EPP) (Ceri et al., 1989). For this to work, the body of the rule should *map* to some facts in $\mathcal{F}$. In other words, there must be a substitution of variables that makes the body of R_1 resembles to F_1. Clearly here we have x is substituted by $John$ and y by Tom. For R_1 to be applicable on F_1 we need a homomorphism that maps $body(R_1)$ to F_1. Let us see the formal definition of rule application.

Definition 3.2.8 (Rules application Baget et al. (2011b)). *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 σ^{safe} is the safe substitution that replaces existential variables with fresh variables (not introduced before). $\alpha(F, R, \sigma)$ is said to be an **immediate derivation** from F.*

Fresh variables are used to avoid the attribution of already used variables to new facts. This would cause a problem when reapplying the rules on the new facts.

Example 3.2.5. *For instance, consider $R = q(x, y) \rightarrow p(z, y)$ and $F = \{q(A, B), r(A)\}$, R is applicable to F because there is a homomorphism from $\{q(x, y)\}$ to $\{q(A, B), r(A)\}$ that substitutes x by A and y by B. The immediate derivation from F is the fact $F' = \{q(A, B), r(A)\} \cup \{p(w, B)\}$ where w is a fresh variable not introduced before.*

Rules can be applied according to an order. For instance, it is possible that a rule, say R_2, is not applicable to any fact in our initial set of facts, but however it becomes so only after applying some other rules. The same thing can happen for other rules. This gives birth to the so-called derivation sequence and $\mathcal{R}$-derivation (Baget et al., 2011b).

Definition 3.2.9 ($\mathcal{R}$-derivation and derivation sequence)**.** *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**) $\langle F_0, ..., F_n \rangle$ such that $F_0 = F$, $F_n = F'$ and for all $0 \leq i < n$ there is a rule $R \in \mathcal{R}$ which is applicable to F_i and F_{i+1} is an immediate derivation from F_i.*

Example 3.2.6. *Let $F = \{q(A, B), r(D), p(x_1, C)\}$ and $\mathcal{R} = \{R_1, R_2\}$ such that:*

- $R_1 = q(x_2, y_1) \wedge r(z_1) \rightarrow d(x_2, z_1)$.
- $R_2 = p(x_3, y_2) \wedge r(z_2) \rightarrow m(z_2, x_3)$.

The following is a derivation sequence:

$$\langle F_0, F_1, F_2 \rangle$$

where:

- $F_0 = F$.
- $F_1 = F \cup \{d(A, D)\}$.
- $F_2 = F_1 \cup \{m(D, x_1)\}$.

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. Note that x_1 is not a fresh variable because z_2 in R_2 is quantified universally.

When we have an initial set of facts $\mathcal{F}$ and a set of rules $\mathcal{R}$, we are interested in *unfolding* all possible knowledge using all possible rules in $\mathcal{R}$. Informally, this can be seen as a saturation mechanism that uses a breadth-first forward chaining scheme. So, We start with a derivation sequence with F_0 being $\mathcal{F}$, then each step i consists of producing a fact, say F_i, from the current fact F_{i-1}, by computing all homomorphisms from the bodies of all rules to F_{i-1}, then performing all corresponding rule applications. The fact F_k obtained after the step k is called the k-saturation of $\mathcal{F}$. Let us formally introduce the concept of k-saturation from Baget et al. (2011b).

Definition 3.2.10 (k-saturation)**.** *Let F be a fact and $\mathcal{R}$ be a set of rules.*

- $\Pi(\mathcal{R}, F)$ *denotes the set of homomorphisms from all rules bodies in $\mathcal{R}$ to F:*
  $\Pi(\mathcal{R}, F) = \{(R, \sigma) \mid R \in \mathcal{R}$ *and σ is a homomorphism from body(R) to F*$\}$
- *The direct saturation of an arbitrary fact F with $\mathcal{R}$ is defined as:*
  $C\ell_{\mathcal{R}}(F) = F \bigcup_{(R,\sigma)\in\Pi(R,F)} \sigma^{safe}(head(R))$.
- *The k-saturation of F with $\mathcal{R}$ is denoted by $C\ell^k_{\mathcal{R}}(F)$ and is inductively defined as follows:*
  $C\ell^0_{\mathcal{R}}(F) = F$ *and, for* $i > 0$, $C\ell^i_{\mathcal{R}}(F) = C\ell_{\mathcal{R}}(C\ell^{i-1}_{\mathcal{R}}(F))$.

We note $C\ell^\infty_{\mathcal{R}}(F) = \bigcup_{k\in\mathbb{N}} C\ell^k_{\mathcal{R}}(F)$, *where* $C\ell^\infty_{\mathcal{R}}(F)$ *is possibly infinite. Please note that $\mathcal{R}$ is a parameter for $C\ell$ like F.*

The following example explains the saturation step by step.

Example 3.2.7. *Consider the following knowledge base* $\mathcal{K} = (\mathcal{F}, \mathcal{R}, \mathcal{N})$*:*

- $\mathcal{F} = \{p(A), q(B), s(B)\}$
- $\mathcal{R} = \{R_1 : p(x) \to r(x, y),\ R_2 : p(x) \wedge s(y) \to p(y),\ R_3 : q(x) \to r(x, y), R_4 : r(x, y) \to t(x)\}$
- $\mathcal{N} = \emptyset$

We start from $\mathcal{F}$:

- $C\ell^0_{\mathcal{R}}(\mathcal{F}) = \mathcal{F}$.
- $C\ell^1_{\mathcal{R}}(\mathcal{F}) = \mathcal{F} \cup \{r(A, y_1), p(B), r(B, y_2)\}$ *using* R_1 *with* $\sigma_1 = \{(x, A)\}$, R_2 *with* $\sigma_2 = \{(x, A), (y, B)\}$ *and* R_3 *with* $\sigma_3 = \{(x, B)\}$.
- $C\ell^2_{\mathcal{R}}(\mathcal{F}) = C\ell^1_{\mathcal{R}}(\mathcal{F}) \cup \{t(A)\}$ *using* R_4 *with* $\sigma_4 = \{(x, A), (y, y_1)\}$ *and* R_4 *with* $\sigma_5 = \{(x, B), (y, y_2)\}$.

After $C\ell^2_{\mathcal{R}}(\mathcal{F})$ *there are no applicable rules that produce new facts.*

As one may see, the saturation procedure halts after 3 steps. If we had the rule $R_5 : p(x) \to m(z, x) \wedge p(x)$ the saturation would never terminate. To see why, think of p as "person" and m as "mother of" then whenever we apply R_5 we get another person y_1 on which we reapply the rule R_5, which in turn gives us another person $y_2, \ldots$ continues *ad infinitum.* Many classes of existential rules have been defined in the literature where some restrictions on the rules are considered to ensure the termination of the saturation procedure (e.g. frontier-guarded rules (Calı et al., 2008)). In this thesis we assume that the set of rules belongs to one of the classes that ensures decidability of entailment (saturation termination). When the saturation procedure halts we refer to $C\ell^{\infty}_{\mathcal{R}}(\mathcal{F})$ as $C\ell^{*}_{\mathcal{R}}(\mathcal{F})$.

The saturation procedure is known as *the naive chase* in the database community (Calı et al., 2008; Abiteboul et al., 1995), it was mainly used to repair a database that does not respect some functional dependencies. We should draw the intention of the reader that the term "chase" is still used in the early works of *Datalog*± (see for instance (Calì et al., 2012; Cali et al., 2010)). Different chases are studied in the literature, e.g. *core chase* (Deutsch et al., 2008), *skolem chase* (Marnette, 2009), etc. the difference between these chases is in their way of handling existential variables and redundancy. For instance, the skolem chase runs a *pre-processing* step where each rule is *skolemized* by replacing each occurrence of an existential variable with a function. Then, the naive chase is applied. As the thesis is not about studying different types of chases, we chose the naive chase (above-explained) as the deduction mechanism.

From a semantics point of view, the saturation operator ($C\ell$) can bee seen as a *fixed-point operator* where $C\ell^{*}_{\mathcal{R}}(\mathcal{F})$ is the least fixed-point of $C\ell$ on $\mathcal{F}$ and $\mathcal{R}$. For instance, $C\ell^{2}_{\mathcal{R}}(\mathcal{F})$ in the previous example is the fixed point of $C\ell$ because $C\ell^{3}_{\mathcal{R}}(\mathcal{F}) = C\ell^{2}_{\mathcal{R}}(\mathcal{F})$, $C\ell^{4}_{\mathcal{R}}(\mathcal{F}) = C\ell^{3}_{\mathcal{R}}(\mathcal{F})$, so on and so forth.

From a model-theoretic perspective, we can see the set of rules as a set of first-order sentences (i.e. a first-order theory) that describes a specific "world". Since a set of sentences may describe infinitely many models it is hard to conclude whether a given fact is a logical consequence of our theory. Therefore, to be able to say for certain that a given fact is a logical consequence of the theory we need to be sure that the fact is true in all possible models of the theory. One can achieve this by the construction of the so-called universal model. This model is particularly interesting because it can be mapped to any other model of our theory. So to check whether a fact F is a logical consequence of theory, it is sufficient to check if its universal model is also a model of F. The saturation procedure described above does construct the universal model. The following theorem of Baget et al. (2011b) shows that.

Theorem 3.2.1 (Saturation). Let F and F' be two facts and $\mathcal{R}$ be a set of rules. Then $F, \mathcal{R} \models F'$ if and only if there is a homomorphism from F' to $C\ell^{*}_{\mathcal{R}}(\mathcal{F})$.

The theorem can be stated equivalently as:

$$F, \mathcal{R} \models F' \textit{ if and only if } C\ell^*_{\mathcal{R}}(F) \models F'$$

From a logical point of view, the operator $C\ell$ can be considered as a *closure operator* or a *consequence operator*. Since this terminology is widely used in logic-based argumentation (e.g. Amgoud and Besnard (2009, 2010, 2013); Amgoud (2014); Vesic (2011)), for convenience we call $C\ell$ a consequence operator and for a given knowledge base $\mathcal{K}$ we interpret $C\ell^*_{\mathcal{R}}(\mathcal{F})$ as the set of all logical consequences of $\mathcal{F}$ and $\mathcal{R}$.

Please note that the term consequence operator is not to be understood in the full classical sense where tautologies are also deduced by the chase procedure. In fact the chase procedure does not compute tautologies, it only applies rules to extend the initial set of facts buy rule firing. Therefore, it computes a part of the all possible logical consequences which have the form of facts.

In the next subsection, we see what happens when the negative constraints are violated and how do we handle such case.

3.2.3 Consistent query answering

Existential rules framework is widely used in Semantic Web and in the so-called Ontology-Based Data Access. Where rules and constraints act as an ontology used to "access" a different data sources. These sources are prone to inconsistencies. In this setting the following assumptions (Lembo et al. (2010)) are made:

Assumption 3.2.2 (Coherence)**.** *The set of rules and negative constraints are satisfiable.*[3]

This assumption is made because in OBDA we assume that the ontology is believed to be reliable as it is the result of a robust construction by domain experts. However, as data can be large and heterogeneous due to merger and fusion, in the OBDA setting the data is assumed to be the source of inconsistency.

Assumption 3.2.3 (Inconsistency)**.** *The set of facts $\mathcal{F}$ may be inconsistent with respect to the rules and negative constraints.*

In what follows we recall the formal definition of *inconsistency* in the existential rules framework, then we introduce different repairing techniques

[3] In logic, a set of formulas is satisfiable if and only if it has at least one model.

which are inspired from the work in database community (Chomicki, 2007) and Description Logics (Lembo et al., 2010; Bienvenu and Rosati, 2013). For such purpose, we introduce the definition of the *repair* of a set of facts and the CQA semantics.

Definition 3.2.11 (Inconsistency). *A set of facts $\mathcal{F}$ is inconsistent with respect to a set of rules $\mathcal{R}$ and negative constraints $\mathcal{N}$ (or inconsistent for short) if and only if $C\ell^*_{\mathcal{R}}(\mathcal{F}) \models body(N)$ such that $N \in \mathcal{N}$.*

This means that the set of facts *violates* the negative constraint N or triggers it. Correspondingly, a knowledge base $\mathcal{K} = (\mathcal{F}, \mathcal{R}, \mathcal{N})$ is inconsistent (with respect to $\mathcal{R}$ and $\mathcal{N}$) if and only if there exists a set of facts $\mathcal{F}' \subseteq \mathcal{F}$ such that $\mathcal{F}'$ is inconsistent. An alternative writing is $C\ell^*_{\mathcal{R}}(\mathcal{F}) \models \bot$. This is straightforward since negative constraints are special forms of rules where falsity $\bot$ can be deduced. Because if the body of a negative constraint is entailed from $C\ell^*_{\mathcal{R}}(\mathcal{F})$ then necessarily $\bot$ is entailed from $C\ell^*_{\mathcal{R}}(\mathcal{F})$.

Example 3.2.8. *Let us consider the following knowledge base $\mathcal{K}$ with: $\mathcal{F} = \{cat(Tom), bark(Tom)\}$, $\mathcal{R} = \{R_1 : cat(x_1) \rightarrow miaw(x_1)\}$, $\mathcal{N} = \{N_1 : bark(x_2) \wedge miaw(x_2) \rightarrow \bot\}$. The saturation yields $C\ell^*_{\mathcal{R}}(\mathcal{F}) = \{cat(Tom), bark(Tom), miaw(Tom)\}$. Observe that this knowledge base violates the negative constraint N_1.*

One way in classical logic to cope with inconsistency is to construct *maximal consistent subset* of the knowledge base that is consistent (Rescher and Manor, 1970). This corresponds to "Data Repairs" (Arenas et al., 1999). Informally, a data repair of a knowledge base $\mathcal{K} = (\mathcal{F}, \mathcal{R}, \mathcal{N})$ is a set of facts $\mathcal{F}'$ such that $\mathcal{F}'$ is consistent and there does not exist a subset of $\mathcal{F}$ that strictly contains $\mathcal{F}'$ that is consistent (Lembo et al., 2010).

Definition 3.2.12 (Repair). *Let $\mathcal{K} = (\mathcal{F}, \mathcal{R}, \mathcal{N})$ be a knowledge base. A data repair (repair for short) of $\mathcal{K}$ is a set of facts $\mathcal{F}' \subset \mathcal{F}$ such that:*

- *$C\ell^*_{\mathcal{R}}(\mathcal{F}') \not\models \bot$ (consistency).*
- *$\forall \mathcal{X} \subset \mathcal{F} \setminus \mathcal{F}'$, $\mathcal{F}' \cup \mathcal{X}$ is inconsistent (maximality).*

Since repairs are computed exclusively on the set of facts and given that the factual part of the knowledge base is the only source of inconsistency we, from now on, abuse slightly the notation and refer to $\mathcal{K}'$ by its set of facts $\mathcal{F}'$. The set of all repairs of $\mathcal{K}$ is denoted as:

$\mathcal{R}epair(\mathcal{K}) = \{\mathcal{F}' \mid \mathcal{F}' \subseteq \mathcal{F}$ *and* $\mathcal{F}'$*respects Definition 3.2.12*$\}$.

Example 3.2.9 (Cont'd Example 3.2.8). *The two possible repairs are:* $\mathcal{P}_1 = \{cat(Tom)\}$ *and* $\mathcal{P}_2 = \{bark(Tom)\}$. *If we consider* $\mathcal{F} \cup \{animal(Tom)\}$, *then the repairs become:*

$$\mathcal{P}_1 = \{cat(Tom), animal(Tom)\} \text{ and } \mathcal{P}_2 = \{bark(Tom), animal(Tom)\}$$

Once all repairs are computed, there are different ways to compute queries that follow from an inconsistent knowledge base. The most prominent way is to allow the entailment of a query if it is entailed from *all repairs*. This is called the CQA semantics or $\mathcal{AR}$*-semantics*. Please note that a query under CQA has either a yes or a no answer. When we say a query is accepted that means it has a yes answer (entailed), otherwise it has a no answer (not entailed). The queries we are considering are ground BCQs where only constants are allowed in the query.

Definition 3.2.13 (CQA semantics). *Let* $\mathcal{K} = (\mathcal{F}, \mathcal{R}, \mathcal{N})$ *be a knowledge base and let* $\mathcal{Q}$ *be a query. Then* $\mathcal{Q}$ *is accepted under CQA in* $\mathcal{K}$, *written* $\mathcal{K} \models_{\mathcal{CQA}} \mathcal{Q}$ *iff for every repair* $\mathcal{P} \in \mathcal{R}epair(\mathcal{K})$, *it holds that* $C\ell^*_{\mathcal{R}}(\mathcal{P}) \models \mathcal{Q}$.

Example 3.2.10 (Cont'd). $C\ell^*_{\mathcal{R}}(\mathcal{P}_1) = \{cat(Tom), animal(Tom), miaw(Tom)\}$, $C\ell^*_{\mathcal{R}}(\mathcal{P}_2) = \{bark(Tom), animal(Tom)\}$. *It is the case that* $\mathcal{K} \models_{\mathcal{CQA}} animal(Tom)$ *but it is not the case that* $\mathcal{K} \models_{\mathcal{CQA}} miaw(Tom)$. *Because* $miaw(Tom)$ *is not entailed from* $\mathcal{P}_2$.

Inconsistency handling in inconsistent knowledge bases can use another concept called *minimal conflicts*. Given a knowledge base $\mathcal{K}$, a set of facts C is called a *minimal conflict* of $\mathcal{K}$ if and only if C is inconsistent and every subset of C is consistent.

Definition 3.2.14 (Minimal conflicts). *Let* $\mathcal{K} = (\mathcal{F}, \mathcal{R}, \mathcal{N})$ *be an inconsistent knowledge base. A set of facts* C *is called a minimal conflict of* $\mathcal{K}$ *if and only if:*

- $C\ell^*_{\mathcal{R}}(C) \models \perp$ *(inconsistency).*
- $\forall X \subset C$, $C \setminus X$ *is consistent (minimality).*

We denote by `conflicts`$(\mathcal{K})$ *the set of all minimal conflicts of* $\mathcal{K}$.

Example 3.2.11 (Conflicts). *Consider the following knowledge base* $\mathcal{K} = (\mathcal{F}, \mathcal{R}, \mathcal{N})$:

- $\mathcal{F} = \{p(A, A), p(B, C), q(C, B), r(C), w(D)\}$.
- $\mathcal{R} = \{q(x, y) \rightarrow s(x, y)\}$.

- $\mathcal{N} = \{p(x,x) \rightarrow \bot, p(x,y) \wedge q(y,x) \wedge r(y) \rightarrow \bot\}$.

We have the following `conflicts`$(\mathcal{K})$*:*

- $C_1 = \{p(A,A)\}$ *and* $C_2 = \{p(B,C), q(C,B), r(C)\}$.

Note that $\{p(B,C), s(C,B), r(C)\}$ *is not a conflict because it is consistent. It is clear from this knowledge base that every fact in* $\mathcal{F}$ *is involved in some inconsistencies except for* $w(D)$*. This means that* $w(D)$ *will be in all repairs. Whereas* $p(B,C)$ *will never succeed to be in a repair that contains* $q(C,B)$ *therefore it will never be in all repairs, hence not accepted under CQA semantics. Here are the repairs to make the example more clear:*

- $\mathcal{P}_1 = \{p(B,C), r(C), w(D)\}$.
- $\mathcal{P}_2 = \{p(B,C), q(C,B), w(D),\}$.
- $\mathcal{P}_3 = \{q(C,B), r(C), w(D),\}$.

We have seen how the CQA semantics handles inconsistency in the existential rules framework. In the next section we introduce logic-based argumentation in existential rules which is another method to handle inconsistency.

3.3 Instantiating Dung's Abstract Framework

Croitoru and Vesic (2013) have presented the first logic-based instantiation in the framework of existential rules.[4] Therefore we recall their instantiation with certain syntactical alteration which will be explained later. Our contribution with respect to this instantiation lays in:

- Providing a fine-grained analysis of the outputs (Subsection 3.3.2).
- Introducing and studying the concept of One-shot Argument-based Explanations (Subsection 3.3.3).
- Studying the different properties that this logic-based instantiation satisfies (Subsection 3.3.4).

[4] In fact, Martinez et al. (2014) have done so but with a slightly different approach. They have extended the language with *defeasible rules*, which makes it fall into the DeLP approaches.

Logic-based instantiations may behave in unexpected ways leading to inconsistent results. There have been lots of works to define *rationality postulates* for logic-argumentation, notably in Caminada and Amgoud (2007); Gorogiannis and Hunter (2011); Amgoud (2014). Overall, these works give some postulates that a logic-based instantiation *should* satisfy to avoid inconsistent results. The followings are the recent postulates proposed in Amgoud (2014).

(`Rationality postulates` Amgoud (2014))

- (P1) *Closure under $C\ell$*: the set of conclusions that can be drawn from any extension should be closed under $C\ell$.
- (P2) *Closure under sub-arguments*: if an argument is accepted in an extension then so are all its sub-arguments. An argument is a sub-argument of another argument if the support of the former is a subset of the support of the latter.
- (P3) *Consistency*: the set of conclusions that can be drawn from any extension should be consistent.
- (P4) *Exhaustiveness*: if an argument is accepted in an extension, then all its sub-parts should also be accepted in that extension
- (P5) *Free precedence*: any argument that is built only from the inconsistency-free part of the knowledge base it should be in every extension.

Croitoru and Vesic (2013) have proven that their instantiation satisfies the rationality postulates of Caminada and Amgoud (2007). Recently, a more general set of postulates have been proposed in Amgoud (2014). In Subsection 3.3.5 we give positive results with respect to the satisfaction of the remaining postulates.

3.3.1 Arguments and attack

Classically, an argument is composed of premises and a conclusion (Besnard and Hunter (2008)). The set of premises is seen as a *justification*, a *support*, a *reason* or a *proof* for the conclusion. We follow this classical definition and alter the definition of Croitoru and Vesic (2013) as follows.

Definition 3.3.1 (Argument). *Given an inconsistent knowledge* $\mathcal{K} = (\mathcal{F}, \mathcal{R}, \mathcal{N})$. *An argument is a tuple* $a = (H, C)$ *such that:*

1. $H \subseteq \mathcal{F}$ *and* $C\ell^*_{\mathcal{R}}(H) \not\models \bot$ *(consistency).*
2. $C = \alpha_0 \wedge \ldots \wedge \alpha_n$ *is a conjunct such that* $\{\alpha_0, \ldots, \alpha_n\} \subseteq C\ell^*_{\mathcal{R}}(H)$ *(entailment).*

The support (resp. conclusion[5]*) of an argument* a *is denoted as* $\texttt{Supp}(a) = H$ *(resp.* $\texttt{Conc}(a) = C$*).*

An argument is defined as tuple in which the support is a set of facts responsible from the entailment of the conclusion C from the knowledge base $\mathcal{K}$. The first clause ensures that the support is consistent which is an important property (Besnard and Hunter (2008)). It is not hard to see that an argument with an inconsistent support will never be accepted in an argumentation process. The second clause ensures the preservation of entailment from the support H to the conclusion C. Note that here arguments are constructed only from the factual part $\mathcal{F}$ of the knowledge base, and there are no rules or negative constraints in the support or the conclusion. The reason to exclude such formulas is due to the Coherence Assumption (page 45) which dictates that the rules and the negative constraints are satisfiable and reliable, therefore they will not be subjects of any attack. Note also that minimality constraint is not imposed, we shall return to this case later.

We should point out that in Croitoru and Vesic (2013) an argument is defined as a derivation sequence (in the sense of Definition 3.2.9, page 42). This way of defining arguments produces unnecessarily large set of arguments as it allows syntactically identical arguments. Let H, C, F_1, F_2, F'_1 and F'_2 be distinct facts. The Croitoru-Vesic arguments (H, F_1, F_2, C) and (H, F'_1, F'_2, C) are considered different despite having the same support and conclusion. This occurs due to the dissimilarity within the derivation sequence. Moreover, the intermediate facts F_1, F_2, F'_1, F'_2 have no effect on the output of the argumentation system as the attack relation is defined with respect to the conclusion and the support (assumption attack as we will see). However, this definition can be found useful in understanding the link between the support and the conclusion. Clearly, with our definition we will definitely get fewer arguments while maintaining the same output.

Arguments may attack each other, different types of attacks are identified in the literature in Besnard and Hunter (2014). *Rebuttal* is a type

[5]Note that the conclusion of an argument is a fact, therefore it can bee seen as a *query* as defined in Section 3.2. From now on, we may use interchangeably the words “claim' or “conclusion” to mean the same thing, the word “query” may also be used instead.

of attack whose source is inconsistency between conclusions. Another well-know type of attack is *undercut*. In this type of attack, the attack relation is defined with respect to the presence of inconsistency between the conclusion of the attacking argument and the support of attacked argument. Direct undercut is a special attack of undercut where the conclusion of the attacking argument is inconsistent with a single atom in the support of the attacked argument. For a full taxonomy of types of attacks see (Besnard and Hunter, 2014, 2008).

In this thesis we focus on the so-called *direct undercut* (*assumption attack*). The reason to avoid classical undercut and rebuttal is because they lead to undesirable results as they give symmetric argumentation frameworks (Amgoud and Besnard (2009)). Moreover, direct undercut provides equivalence results between extensions and CQA semantics studied in Subsection 3.2.3, this would allow us to use argumentation to explain query answering under inconsistency as we shall see.

Definition 3.3.2 (Attack). *An argument a attacks b if and only if $\exists h \in \mathtt{Supp}(b)$ such that $C\ell^*_{\mathcal{R}}(\{\mathtt{Conc}(a), h\}) \models \bot$.*

In fact this particularity makes the attack relation not symmetric as shown in Croitoru and Vesic (2013).

Proposition 3.3.1 (Non-symmetry). Let $\mathcal{K} = (\mathcal{F}, \mathcal{R}, \mathcal{N})$ be a knowledge base, $\mathcal{H} = (\mathcal{A}, \mathcal{X})$ be its corresponding argumentation framework. If $\mathcal{K} = (\mathcal{F}, \mathcal{R}, \mathcal{N})$ is inconsistent and $\mathcal{X}$ is not empty then $\mathcal{X}$ is not symmetry and irreflexive.

To show the non-symmetry of the attack consider the following example.

Example 3.3.1 (Non-symmetry). *Let $\mathcal{F} = \{p(M), q(M), r(M)\}$, $\mathcal{R} = \emptyset$, $\mathcal{N} = \{p(x) \wedge q(x) \wedge r(x) \rightarrow \bot)\}$. Let $a = (\{p(M), q(M)\}, p(M) \wedge q(M))$, $b = (\{r(M)\}, r(M))$. We have $(a, b) \in \mathcal{X}$ and $(b, a) \notin \mathcal{X}$ because there exists no $h \in \mathtt{Supp}(a)$ which is inconsistent with $\mathtt{Conc}(b) = r(M)$. However, the set $\{p(M), q(M)\}$ is indeed inconsistent with $r(M)$, but according to the definition of the attack we consider a single atom in the support of a. In fact, if the attack were to be defined as $H \subseteq \mathtt{Supp}(a)$ then b would definitely attack a. This type of attack is called classical undercut.*

This attack relation is also irreflexive. This is guaranteed by the fact that for any argument a, $\mathtt{Supp}(a)$ is consistent and it entails $\mathtt{Conc}(a)$, therefore $\mathtt{Supp}(a)$ and $\mathtt{Conc}(a)$ are consistent together. Consequently, an argument cannot attack itself.

Another case which can occur is the emptiness of the attack relation. It occurs when all minimal conflicts in the knowledge base are unary. That means the only inconsistent facts in the knowledge base are those which are self-contradictory. Hence, none of them will construct an argument. Consequently there will be no attack. This leads us to the following.

Proposition 3.3.2 (Emptiness). If $\mathcal{K}$ has only unary minimal conflicts then $\mathcal{X}$ is empty.

Example 3.3.2.

Example 3.3.3. *Consider* $\mathcal{F} = \{p(M), r(M)\}$, $\mathcal{R} = \emptyset$, $\mathcal{N} = \{p(X) \rightarrow \bot)\}$. *We have only one argument in this case* $a = (\{r(M)\}, r(M))$. *It is clear that* $p(M)$ *is self-contradictory because it triggers the negative constraint. It is not hard to see that the attack relation is empty.*

Emptiness can occur in other cases. When the knowledge base is consistent or the set of arguments is empty due to the emptiness of the set of facts then the attack relation would be empty.

Before completing the definition of the argumentation framework let us introduce some important notations.

Notation 3.3.1. Let $\mathcal{K}$ be a knowledge base, $F \subseteq \mathcal{F}$ be a set of facts and S be a set of arguments. We adapt the following notations:

- $\mathtt{Args}(F) = \{a \mid a \textit{ is an argument such that } \mathtt{Supp}(a) \subseteq F\}$. This refers to the set of all possible arguments that can be constructed from a given set of facts F.

- $\mathtt{Base}(S) = \bigcup \mathtt{Supp}(a_i)$ such that $a_i \in S$. A base of a set of arguments is a set of facts that contains all supports of all arguments of S.

- The set of all arguments that can be constructed over $\mathcal{K}$ is denoted as $\mathtt{Args}(\mathcal{F})$.

An argumentation framework is defined as follows (from Croitoru and Vesic (2013)).

Definition 3.3.3 (Argumentation framework)**.** *Let* $\mathcal{K} = (\mathcal{F}, \mathcal{R}, \mathcal{N})$ *be a knowledge base. The corresponding argumentation framework is a pair* $\mathcal{H} = (\mathtt{Args}(\mathcal{F}), \mathcal{X})$ *where* $\mathtt{Args}(\mathcal{F})$ *is the set of all arguments that can be constructed from* $\mathcal{F}$ *and* $\mathcal{X}$ *is the corresponding attack relation as specified in Definition 3.3.2.*

For an argumentation framework $\mathcal{H} = (\mathcal{A}, \mathcal{X})$ we denote by $\mathtt{Ext}_x(\mathcal{H})$ the set of its extensions with respect to the semantics x. We use the abbreviations c, p, s, and g for respectively complete, preferred, stable and grounded semantics. We denote by $\mathtt{Sc}_x(\mathcal{H})$ and $\mathtt{Cr}_x(\mathcal{H})$ the set of all arguments that are skeptically accepted and credulously accepted in $\mathcal{H}$ under the semantics x respectively.

Let us take an example of how we construct an argumentation framework from a given knowledge base.

Example 3.3.4 (Pick two!)**.** *Consider the "Fast, Good or Cheap. Pick two!" project management principle. It states the fact that the three properties Fast, Good and Cheap of a project are interrelated, and it is not possible to optimize all the three, then one should always pick two of the three.*

Figure 3.1: Pick any two.

It can be represented as a knowledge base as follows:

$\mathcal{F} = \{project(P), isfast(P), isgood(p), ischeap(P)\}$

$\mathcal{R} = \{isfast(x) \land isgood(x) \rightarrow isexpensive(x)\}$

$\mathcal{N} = \{ischeap(x) \land isexpensive(x) \rightarrow \bot\}$

Some of the arguments that can be generated from $\mathcal{K}$ are presented in Table 3.1.[6] *The attacks are defined with respect to the following set of conflicts:*

$$\begin{gathered}\mathtt{conflicts}(\mathcal{K}) = \{C_1, C_2\} \textit{ such that} \\ C_1 = \{ischeap(P), isfast(P), isgood(P)\} \textit{ and} \\ C_2 = \{ischeap(P), isexpensive(P)\}.\end{gathered}$$

The attack relation is presented in the argumentation framework in Figure 3.2. The blue edges represent the arguments attacked by the extension

[6]For space reasons the set of all arguments are not provided here. Please see Appendix A of Chapter 3, Example 3.3.4.

$a_3 = (\{project(P), isgood(P), ischeap(P)\}, project(P) \wedge isgood(P) \wedge ischeap(P))$
$a_7 = (\{project(P), isgood(P), ischeap(P)\}, project(P) \wedge isgood(P))$
$a_8 = (\{project(P), isgood(P), ischeap(P)\}, project(P) \wedge ischeap(P))$
$a_9 = (\{project(P), isgood(P), ischeap(P)\}, isgood(P) \wedge ischeap(P))$
$a_{16} = (\{project(P), isgood(P), ischeap(P)\}, project(P))$
$a_{17} = (\{project(P), isgood(P), ischeap(P)\}, isgood(P))$
$a_{18} = (\{project(P), isgood(P), ischeap(P)\}, ischeap(P))$
$a_{23} = (\{project(P), isgood(P)\}, project(P) \wedge isgood(P))$
$a_{24} = (\{isgood(P), ischeap(P)\}, isgood(P) \wedge ischeap(P))$
$a_{25} = (\{project(P), ischeap(P)\}, project(P) \wedge ischeap(P))$
$a_{28} = (\{project(P), isgood(P)\}, project(P))$
$a_{29} = (\{project(P), isgood(P)\}, isgood(P))$
$a_{30} = (\{isgood(P), ischeap(P)\}, isgood(P))$
$a_{31} = (\{isgood(P), ischeap(P)\}, ischeap(P))$
$a_{34} = (\{project(P), ischeap(P)\}, project(P))$
$a_{35} = (\{project(P), ischeap(P)\}, ischeap(P))$
$a_{40} = (\{project(P)\}, project(P))$
$a_{42} = (\{ischeap(P)\}, ischeap(P))$
$a_{43} = (\{isgood(P)\}, isgood(P))$

Table 3.1: The arguments of $\mathcal{E}_3$. There is 54 arguments in $\mathcal{A}$.

$\mathcal{E}_3$ *(*$\texttt{range}^+(\mathcal{E}_3)$*). In what follows we present the extensions under the stable/preferred semantics:*

- $\mathcal{E}_1 = \{a_2, a_{10}, a_{11}, a_{12}, a_{13}, a_{14}, a_{15}, a_{22}, a_{23}, a_{27}, a_{28}, a_{29}, a_{32}, a_{33}, a_{38}, a_{39}, a_{40}, a_{41}, a_{43}, a_{44}, a_{45}, a_{46}, a_{47}, a_{48}, a_{49}, a_{50}, a_{51}, a_{52}, a_{53}, a_{54}\}$.

- $\mathcal{E}_2 = \{a_1, a_4, a_5, a_6, a_{19}, a_{20}, a_{21}, a_{22}, a_{25}, a_{26}, a_{32}, a_{33}, a_{34}, a_{35}, a_{36}, a_{37}, a_{40}, a_{41}, a_{42}, a_{44}\}$.

- $\mathcal{E}_3 = \{a_3, a_7, a_8, a_9, a_{16}, a_{17}, a_{18}, a_{23}, a_{24}, a_{25}, a_{28}, a_{29}, a_{30}, a_{31}, a_{34}, a_{35}, a_{40}, a_{42}, a_{43}\}$.

For this argumentation framework $\texttt{Sc}_s(\mathcal{H}) = \texttt{Sc}_p(\mathcal{H}) = \{a_{40}\}$ *which corresponds to the grounded extension. The credulous arguments* $\texttt{Cr}_s(\mathcal{H}) = \texttt{Cr}_p(\mathcal{H}) = \mathcal{A}$*. This means that there are no rejected arguments.*

It is easy to check that the argumentation framework is asymmetric. The argument $a_1 = (\{project(P), isfast(P), ischeap(P)\}, project(P) \wedge isfast(P) \wedge ischeap(P))$ attacks a_7 because it is impossible to have a fast and good project on the one hand and a cheap project on the other hand since the former gives an expensive project. However, a_7 does not attack a_1.

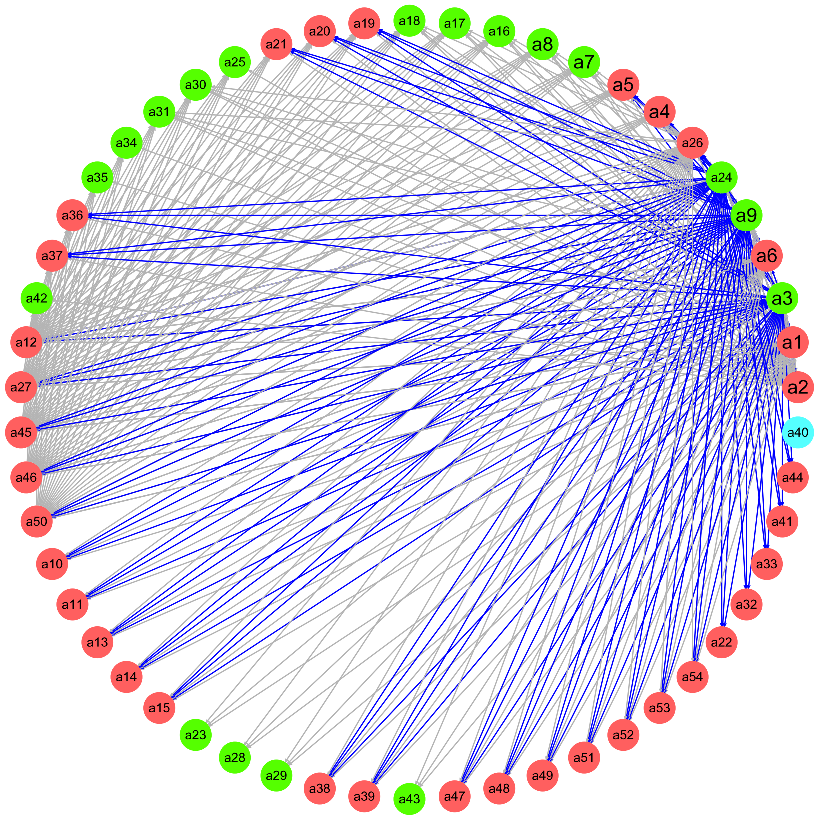


Figure 3.2: The argumentation framework. Green vertices belong to extension $\mathcal{E}_3$. The blue edges represent $\texttt{range}^+(\mathcal{E}_3)$ where it shows the conflict-freeness and defense of $\mathcal{E}_3$. The blue vertex is the skeptically accepted argument.

3.3.2 Outputs of logic-based argumentation

Besides the conventional output of abstract argumentation frameworks shown in Chapter 2, Section 2.2 and the justification state in Section 2.3. Logic-based argumentation frameworks allow to exploit the structure of arguments to reason in terms of acceptable conclusions.

The set of plausible conclusions of an argumentation framework are those conclusions that can be inferred from all extensions under a given semantics (Amgoud (2014)).

Definition 3.3.4 (Plausible conclusions)**.** *Let $\mathcal{K} = (\mathcal{F}, \mathcal{R}, \mathcal{N})$ be a knowledge base and $\mathcal{H} = (\mathcal{A}, \mathcal{X})$ its corresponding argumentation framework. The set* *of plausible conclusions of $\mathcal{H} = (\mathcal{A}, \mathcal{X})$ under a semantics x is defined as:*

$$\mathtt{Output}_x(\mathcal{H}) = \begin{cases} \bigcap\limits_{\mathcal{E} \in \mathtt{Ext}_x(\mathcal{H})} \mathtt{Concs}(\mathcal{E}) & \textit{if } \mathtt{Ext}_x(\mathcal{H}) \neq \emptyset \\ \emptyset & \textit{Otherwise} \end{cases}$$

Such that $\mathtt{Concs}(\mathcal{E}) = \{\mathtt{Conc}(a) \mid a \in \mathcal{E}\}$.

At first glance the plausible conclusions seem to be those facts which are the conclusion of skeptically accepted arguments. In fact this is not always the case, to see why we let us first introduce some notions.

Definition 3.3.5 (Outputs)**.** *Let $\mathcal{K} = (\mathcal{F}, \mathcal{R}, \mathcal{N})$ be a knowledge base, $\mathcal{H} = (\mathcal{A}, \mathcal{X})$ be its corresponding argumentation framework. We distinguish the following outputs with respect to a semantics x.*

- *The skeptical output:* $\mathtt{Output}_x^{sc}(\mathcal{H}) = \{\mathtt{Conc}(a) \mid a \in \mathtt{Sc}_x(\mathcal{H})\}$.
- *The credulous output:* $\mathtt{Output}_x^{cr}(\mathcal{H}) = \{\mathtt{Conc}(a) \mid a \in \mathtt{Cr}_x(\mathcal{H})\}$.
- *The rejected output:* $\mathtt{Output}_x^{re}(\mathcal{H}) = \{\mathtt{Conc}(a) \mid a \notin \mathtt{Cr}_x(\mathcal{H})\}$.

The skeptical output is the conclusions of all skeptical arguments. The credulous output are those of credulous arguments and the rejected output are those of the rejected arguments. It is clear that credulous output contains all conclusions except of those of the rejected arguments.

Fact 3.3.1. $\mathtt{Output}_x^{sc}(\mathcal{H}) \subseteq \mathtt{Output}_x(\mathcal{H}) \subseteq \mathtt{Output}_x^{cr}(\mathcal{H})$.

However, the statement $\mathtt{Output}_x^{sc}(\mathcal{H}) \subseteq \mathtt{Output}_x(\mathcal{H})$ means that there are some conclusions which could be in $\mathtt{Output}_x(\mathcal{H})$ but not skeptical. They are not strong enough to be skeptical and not weak as credulous. To show why this happens, consider the following example.

Example 3.3.5. *The following is an inconsistent knowledge base $\mathcal{K}$ where:*

- $\mathcal{F} = \{jaguar(T), leopard(T)\}$.
- $\mathcal{R} = \{jaguar(x) \rightarrow animal(x), leopard(x) \rightarrow animal(x)\}$.
- $\mathcal{N} = \{jaguar(x) \wedge leopard(x) \rightarrow \bot\}$.

We have the following $\mathtt{Args}(\mathcal{F})$*:*

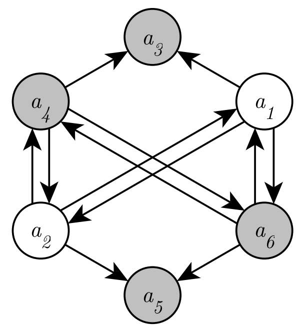


Figure 3.3: Plausible conclusions are not always skeptical. The gray-colored arguments are the supporting arguments of the query $\mathcal{Q} = animal(T)$.

$$a_1 = (\{jaguar(T)\}, jaguar(T)) \qquad a_2 = (\{leopard(T)\}, leopard(T))$$
$$a_3 = (\{leopard(T)\}, animal(T)) \quad a_4 = (\{jaguar(T)\}, animal(T) \wedge jaguar(T))$$
$$a_5 = (\{jaguar(T)\}, animal(T)) \quad a_6 = (\{leopard(T)\}, animal(T) \wedge leopard(T))$$

The attack relation is drawn in Figure 3.3. Consequently, we get the following $\mathtt{Ext}_x(\mathcal{H})$ *such that* $x \in \{s, p\}$*:*

$$\mathcal{E}_1 = \{a_1, a_4, a_5\} \quad \mathcal{E}_2 = \{a_2, a_3, a_6\}$$

We have the following outputs: $\mathtt{Sc}_x(\mathcal{H}) = \emptyset$, thus $\mathtt{Output}_x^{sc}(\mathcal{H}) = \emptyset$. But still $\mathtt{Output}_x(\mathcal{H}) = \{animal(T)\}$. Note that $\mathtt{Output}_x^{cr}(\mathcal{H}) = \mathcal{F} \cup \{animal(T)\}$.

As one can see, no argument is skeptically accepted but yet the conclusion $\mathcal{Q} = animal(T)$ is plausible. This is due to the fact that $\mathcal{Q}$ is inferred from all extensions by means of different arguments.

In order to capture such subtle difference let us define the notion of a supporting argument. An argument a supports a query if and only if the query is the conclusion of the argument a.

Definition 3.3.6 (Support)**.** *Let* $\mathcal{H} = (\mathcal{A}, \mathcal{X})$ *be an argumentation framework over an inconsistent knowledge base* $\mathcal{K}$ *and let* $\mathcal{Q}$ *be a query. An argument* $a \in \mathcal{A}$ *supports the query* $\mathcal{Q}$ *if and only if* $\mathtt{Conc}(a) = \mathcal{Q}$*. We call* a ***a supporter*** *or* ***a supporting argument*** *of* $\mathcal{Q}$*. The set of all supporters of a given query* $\mathcal{Q}$ *is denoted as* $\mathtt{S}(\mathcal{Q})$*.*

After introducing the notion of supporting arguments we can clarify the issue of Example 3.3.5.

Definition 3.3.7 (Universal output)**.** *Let $\mathcal{K} = (\mathcal{F}, \mathcal{R}, \mathcal{N})$ be a knowledge base, $\mathcal{H} = (\mathcal{A}, \mathcal{X})$ be the corresponding argumentation framework and $F \in C\ell^*_{\mathcal{R}}(\mathcal{F})$. The universal output of $\mathcal{H}$ is defined as follows:*

$$\texttt{Output}^{un}_x(\mathcal{H}) = \{F \mid F \in \texttt{Output}^{cr}_x(\mathcal{H}) \textit{ and } \forall \mathcal{E} \in \texttt{Ext}_x(\mathcal{H}), \exists a \in \texttt{S}(F) \textit{ such that } a \in \mathcal{E}\}.$$

This output corresponds to those conclusions that have supporting arguments which are distributed over all the extensions. It is clear that the skeptical output is included in the universal output.

Fact 3.3.2. $\texttt{Output}^{sc}_x(\mathcal{H}) \subseteq \texttt{Output}^{un}_x(\mathcal{H})$.

Having the different outputs of the logic-based argumentation framework, we can now define the acceptance of a query (or conclusion) as follows.

Definition 3.3.8 (Query acceptance)**.** *Let $\mathcal{H} = (\mathcal{A}, \mathcal{X})$ be an argumentation framework built over an arbitrary inconsistent knowledge base, $\mathcal{Q}$ be a query and x be a semantics. Then, $\mathcal{Q}$ is:*

- *Skeptically accepted under x if and only if* $\texttt{Output}^{sc}_x(\mathcal{H}) \models \mathcal{Q}$.
- *Universally accepted under x if and only if* $\texttt{Output}^{un}_x(\mathcal{H}) \models \mathcal{Q}$.
- *Credulously accepted under x if and only if* $\texttt{Output}^{cr}_x(\mathcal{H}) \models \mathcal{Q}$.
- *Rejected under x if and only if* $\texttt{Output}^{re}_x(\mathcal{H}) \models \mathcal{Q}$.

Corollary 3.3.1. If $\mathcal{Q}$ is skeptically accepted then it is universally accepted. The converse is false.

Let us see this on an example.

Example 3.3.6 (Cont'd Example 3.3.5)**.** *Let us consider $\mathcal{F}' = \mathcal{F} \cup \{felidae(T)\}$. We get* $\texttt{Args}(\mathcal{F}') = \texttt{Args}(\mathcal{F})$ *plus:*

$a_7 = (\{jaguar(T), felidae(T)\}, jaguar(T))$

$a_8 = (\{jaguar(T), felidae(T)\}, jaguar(T) \wedge felidae(T))$

$a_9 = (\{jaguar(T), felidae(T)\}, felidae(T) \wedge animal(T))$

$a_{10} = (\{jaguar(T), felidae(T)\}, jaguar(T) \wedge felidae(T) \wedge animal(T))$

$a_{11} = (\{jaguar(T), felidae(T)\}, animal(T))$

$a_{12} = (\{leopard(T), felidae(T)\}, leopard(T))$

$a_{13} = (\{leopard(T), felidae(T)\}, leopard(T) \wedge felidae(T))$

$a_{14} = (\{leopard(T), felidae(T)\}, felidae(T) \wedge animal(T))$

$a_{15} = (\{leopard(T), felidae(T)\}, leopard(T) \wedge felidae(T) \wedge animal(T))$

$a_{16} = (\{leopard(T), felidae(T)\}, animal(T))$

$a_{17} = (\{felidae(T)\}, felidae(T))$

We have the following outputs:

- $\texttt{Output}_x^{sc}(\mathcal{H}) = \{felidae(T)\}$.
- $\texttt{Output}_x^{un}(\mathcal{H}) = \{animal(T), felidae(T) \wedge animal(T)\}$.
- $\texttt{Output}_x(\mathcal{H}) = \{felidae(T), animal(T), felidae(T) \wedge animal(T)\}$.

The query $\mathcal{Q}_1 = felidae(T)$ is skeptically accepted and universally accepted. The query $\mathcal{Q}_2 = felidae(T) \wedge animal(T)$ is universally accepted but not skeptically accepted. The query $\mathcal{Q}_3 = jaguar(T) \wedge felidae(T)$ is credulously accepted.

In the next subsection we characterize the output of logic-based argumentation framework in attempt to give a formal ground for the concept of explanation.

3.3.3 Characterizing the outputs

The universal and non-universal acceptance can be further characterized in a precise way. The goal of introducing such characterization is to be able to understand why a query is universally accepted or not. Consequently, to be able to explain to the users why a query is (non)-universally accepted. Note that we may omit in what follows the subscript that refers to the semantics in the notation $\texttt{Ext}_x(\mathcal{H})$ to mean stable or preferred.

Definition 3.3.9 (Reduct of extension)**.** *Given an extension $\mathcal{E} \subseteq \mathcal{A}$ and a query $\mathcal{Q}$. The reduct $\mathcal{E}^{\mathcal{Q}} \subseteq \mathcal{E}$ of the extension $\mathcal{E}$ w.r.t the query $\mathcal{Q}$ is defined as the non-empty intersection $\texttt{S}(\mathcal{Q}) \bigcap \mathcal{E}$. The reduct of the set of all extensions $\texttt{Ext}(\mathcal{H})$ with respect to $\mathcal{Q}$ is defined as $\texttt{Ext}(\mathcal{H})^{\mathcal{Q}} = \{\mathcal{E}^{\mathcal{Q}} | \mathcal{E} \in \texttt{Ext}(\mathcal{H})\}$.*

The reduct $\mathcal{E}^{\mathcal{Q}}$ of the extension $\mathcal{E}$ with respect to the query $\mathcal{Q}$ is defined as the set of all supporters of $\mathcal{Q}$ which belong to $\mathcal{E}$.

Definition 3.3.10 (Complete reduct)**.** *The set of all reducts* $\texttt{Ext}(\mathcal{H})^{\mathcal{Q}}$ *with respect to a query* $\mathcal{Q}$ *is complete if and only if there exists no* $\mathcal{E} \in \texttt{Ext}(\mathcal{H})$ *such that* $\mathcal{E}^{\mathcal{Q}} \notin \texttt{Ext}(\mathcal{H})^{\mathcal{Q}}$*.*

This means that the set of all reducts is complete if it covers all the extensions.

Example 3.3.7 (Cont'd Example 3.3.5)**.** *Let us see how we compute reducts for the following queries:*

- *Consider* $\mathcal{Q}_1 = animal(T)$ *we get* $\mathcal{E}_1^{\mathcal{Q}_1} = \{a_5, a_4\}$ *and* $\mathcal{E}_2^{\mathcal{Q}_1} = \{a_3, a_6\}$*.*
- *Consider* $\mathcal{Q}_2 = animal(T) \wedge leopard(T)$ *we get* $\mathcal{E}_2^{\mathcal{Q}_2} = \{a_6\}$ *and* $\mathcal{E}_1^{\mathcal{Q}_2}$ *does not exist.*

$\texttt{Ext}(\mathcal{H})^{\mathcal{Q}_1} = \{\mathcal{E}_1^{\mathcal{Q}_1}, \mathcal{E}_2^{\mathcal{Q}_1}\}$ *is complete and* $\texttt{Ext}(\mathcal{H})^{\mathcal{Q}_2} = \{\mathcal{E}_2^{\mathcal{Q}_2}\}$ *is not because the latter does not cover the extension* $\mathcal{E}_1$ *as it has no reduct.*

Using the notion of reducts we can establish the following relations with query acceptance.

Proposition 3.3.3. Given a query $\mathcal{Q}$. Then, the following statements hold:

1. $\texttt{Output}_x^{sc}(\mathcal{H}) \models \mathcal{Q} \Leftrightarrow \texttt{Ext}(\mathcal{H})^{\mathcal{Q}}$ is complete and $\bigcap \texttt{Ext}(\mathcal{H})^{\mathcal{Q}} \neq empty$.
2. $\texttt{Output}_x^{un}(\mathcal{H}) \models \mathcal{Q} \Leftrightarrow \texttt{Ext}(\mathcal{H})^{\mathcal{Q}}$ is complete.
3. $\texttt{Output}_x^{cr}(\mathcal{H}) \models \mathcal{Q} \Leftrightarrow \texttt{Ext}(\mathcal{H})^{\mathcal{Q}} \neq \emptyset$.

   The symbol "$\Leftrightarrow$" refers to equivalence.

The equivalence statements define the three types of acceptance in terms of the reducts of extensions. It is clear in the first statement that the intersection of reducts w.r.t $\mathcal{Q}$ corresponds to all skeptically accepted arguments that support $\mathcal{Q}$. The second statement stipulates that for a query $\mathcal{Q}$ to be universally accepted it has to be supported from all possible extensions (hence the completeness). The third statement dictates that a query $\mathcal{Q}$ is credulously accepted if and only if it has some supporters in some extensions.

Still the characterization of universal acceptance is not precise enough to give a complete account. Consider the set of all reducts w.r.t $\mathcal{Q} = animal(T)$ which is $\{\{a_5, a_4\}, \{a_3, a_6\}\}$ (Example 3.3.7). This set holds sufficient "reasons" to believe that $\mathcal{Q}$ is universally accepted. However, it would have been sufficient to just have $\{\{a_5\},\{a_3\}\}$ or other combinations that keep in a *minimal way* those supporters which *preserve* the existence of the query $\mathcal{Q}$ in each extension.

In combinatorics and diagnosis theory this problem is known as the hitting set problem (Reiter, 1987). It is also referred to as the transversal problem in hypergraph theory (Eiter and Gottlob, 2002).

Definition 3.3.11 (Hitting set). *Given a collection $\mathcal{C} = \{S_1, ..., S_m\}$ of finite nonempty subsets of a set $\mathcal{B}$ (the background set). A* hitting set *of $\mathcal{C}$ is a set $A \subseteq \mathcal{B}$ such that $S_j \cap A \neq \emptyset$ for all $S_j \in \mathcal{C}$. A hitting set of $\mathcal{C}$ is minimal (w.r.t $\subseteq$) if and only if no proper subset of it is a hitting set of $\mathcal{C}$. A minimum hitting set is a minimal hitting set w.r.t set-cardinality.*

Finding one/all minimal/minimum hitting set is an interesting problem. It has a relation with different problems in different areas. In what follows we give a precise characterization of universal acceptance by means of the hitting set problem.

Definition 3.3.12 (Proponent set). *A set of arguments $S \subseteq \mathcal{A}$ is a proponent set of $\mathcal{Q}$ if and only if S is a minimal (w.r.t $\subseteq$) hitting set of $\texttt{Ext}(\mathcal{H})^{\mathcal{Q}}$ and $\texttt{Ext}(\mathcal{H})^{\mathcal{Q}}$ is a complete reduct.*

We get the following characterization which is similar to the concept of a complete base in Thang et al. (2009).

Proposition 3.3.4. $\mathcal{Q}$ is universally accepted $\Leftrightarrow$ $\mathcal{Q}$ has a proponent set.

Example 3.3.8. *$\mathcal{Q} = animal(T)$ has 4 proponent sets $S_1 = \{a_5, a_3\}$, $S_2 = \{a_5, a_6\}$, $S_3 = \{a_4, a_3\}$ and $S_4 = \{a_4, a_6\}$.*

A proponent set holds the smallest set of arguments which are distributed over all extensions and support the query $\mathcal{Q}$. So, if one extension does not contain any supporter then the query is not universally accepted. The reason for the absence of such supporter is what we call the presence of a *block*. We follow the notion of a **block** from Modgil and Caminada (2009b) and instantiate it in our setting. A block B is a set of arguments which are (1) all credulously accepted, (2) attack all the supporters of $\mathcal{Q}$, and (3) they can all together be extended to form an extension.

Definition 3.3.13 (Block). *Let $\mathcal{Q}$ be a query and let $\mathcal{C} = \{\texttt{range}^{-}(a) | a \in \texttt{S}(\mathcal{Q})\}$. A set of arguments $B \subseteq \mathcal{A}$ is a block of $\mathcal{Q}$ if and only if:*

1. *B is a hitting set of $\mathcal{C}$; and,*

2. *There exists an admissible set $B' \subseteq \mathcal{A}$ such that $B \subseteq B'$.*

Note that a query may have more than one block.

Interestingly, a block is related to the hitting set problem. Informally, we take all the supporters of the query $\mathcal{Q}$ and for each supporter we get its attackers (i.e. $\texttt{range}^-(a)$) then we look for those hitting sets (over all sets of attackers) that can be extended to an extension, in other words those which belong to the same admissible set. Note that while a block is necessarily a hitting set it is not necessarily a minimal one.

The last requirement is very important as the following example shows.

Example 3.3.9. *Consider again the query $\mathcal{Q}_1 = animal(T)$ of Example 3.3.7. Let us compute its block(s):*

- $\texttt{S}(\mathcal{Q}_1) = \{a_3, a_4, a_5, a_6\}$.
- $\texttt{range}^+(a_3) = \texttt{range}^+(a_6) = \{a_1, a_4\}$.
- $\texttt{range}^+(a_4) = \texttt{range}^+(a_5) = \{a_2, a_6\}$.

We get the following hitting sets: $\{a_1, a_6\}$,$\{a_1, a_2\}$,$\{a_4, a_6\}$,$\{a_4, a_2\}$. Observe that none of them is considered as a block, because non of them can be extended to form an extension. In fact, they violate the conflict-freeness condition. While $\mathcal{Q}_1$ has no blocks, the query $\mathcal{Q}_2 = animal(T) \wedge leopard(T)$ has two blocks $\{a_1\}$ and $\{a_4\}$.

Since the concept of a block characterizes exactly non-universal acceptance, the following result is straightforward.

Proposition 3.3.5. A query $\mathcal{Q}$ has a block iff it has no proponent set.

Proponent sets and blocks can be seen as causes describing why a query is universally accepted or not. They describe precisely the reasons behind the acceptance or non-acceptance of the query. Moreover, if a query is (not) universally accepted then there is always an explanation (block or proponent set) to explain its state, which is an interesting feature. As block and proponent set are set of arguments we call then *One-shot Argument-based Explanations.*

After characterizing the outputs of argumentation frameworks in the existential rules framework, we shift now to study the general properties of such argumentation frameworks.

3.3.4 Properties

In this subsection we provide some important properties for the class of argumentation we defined in this chapter. We investigate finiteness, coherence, well-foundedness, relative groundedness and non-triviality.

Let us start with the following simple fact.

Proposition 3.3.6. If $\mathcal{K}$ is consistent then the corresponding argumentation framework $\mathcal{H}$ has no attack and it produces only one complete extension $\mathcal{E} = \mathcal{A}$ which is grounded, preferred and stable.

In this thesis we are interested in the class of argumentation frameworks that are built over **inconsistent** knowledge bases in the existential rule framework and whose set of arguments is not empty. This class is denoted as $\text{ARG}_\exists$. When we write $\mathcal{H} \in \text{ARG}_\exists$ we mean that there exists an inconsistent knowledge base $\mathcal{K}$ such that $\mathcal{H}$ is its corresponding argumentation framework.

It turns out that the argumentation frameworks of $\text{ARG}_\exists$ enjoy the finiteness property. Finiteness is divided into two types, (1) finiteness of arguments and (2) finiteness of the set of arguments. First, let us define the notion of a finite argument:

Definition 3.3.14 (Finite argument)**.** *An argument a is finite if and only if* $\texttt{Conc}(a)$ *is finite.*

This means that the conclusion of the argument does not contain infinite conjunctions. Note that, since $\texttt{Supp}(a)$ is constructed from a finite set of facts $\mathcal{F}$, it is finite.

Proposition 3.3.7 (Finiteness). $\forall \mathcal{H} \in \text{ARG}_\exists$ the following hold: (i) $\forall a \in \mathcal{A}$, a is finite; and (ii) $\mathcal{H}$ is finite (thus finitary).

Proof (Sketch)**.** *(i) Let* $\texttt{Conc}(a) = \alpha_0 \wedge \alpha_2 \wedge \ldots$ *be infinite. By definition* $\forall \alpha_i \in \{\alpha_0, \alpha_2, \ldots\}, \alpha_i \in C\ell^*_{\mathcal{R}}(\texttt{Supp}(a))$*. This means* $C\ell^*_{\mathcal{R}}(\texttt{Supp}(a))$ *is infinite. This is in contradiction with the assumption we have made in the logical language section page 43 about the finiteness of the saturation. (ii) if it were the case that $\mathcal{H}$ is not finite then $\mathcal{A}$ should contain infinitely many arguments. This means we have arguments with infinite conclusions. This is not the case from (i).* ∎

The first property ensures that there is no argument that contains infinite conjunctions. This is very important for proving the finiteness of the argumentation framework. Recall that finiteness requires that the set of arguments of a given argumentation framework is finite. If we have no infinite arguments then the set of all arguments is bounded. The second property dictates that argumentation frameworks of $\text{ARG}_\exists$ are finite and finitary. Recall that finitary requires that the set of attackers for each argument is finite. It is obvious that a finite argumentation framework is necessarily a finitary one (Dung, 1995). This is due to the fact that the set of arguments is finite therefore every argument would have a finite set of attackers.

Now we shift to prove the rest of the properties, we shall prove non-triviality. To do so we need to show that there is always a non-empty admissible set in the argumentation frameworks of $\text{ARG}_\exists$. Next, we turn to the property that stipulates that there is no rejected arguments in $\text{ARG}_\exists$, we prove such property by stating that every argument belongs to an admissible set. To prove the two properties we need to show that there exists always an argument that defends itself (guaranties the existence of non-empty admissible set), then we show that every argument in $\text{ARG}_\exists$ either it defends itself or it is defend by another argument that eventually defends itself.

Proposition 3.3.8. $\forall \mathcal{H} \in \text{ARG}_\exists$, the argument $a = (A, \bigwedge A)$ such that A is a maximal consistent set of facts is an admissible set.

Proof (By contradiction)**.** *To show that $\{a\}$ is admissible we need to prove that whenever it is attacked then it defends itself. Let us proceed by contradiction. Assume that there exists an argument b that attacks a, that means there exists $h \in \texttt{Supp}(a)$ such that $\{\texttt{Conc}(b), h\}$ is inconsistent, consequently $\texttt{Supp}(b) \cup \{h\}$ is inconsistent.*

Assume further that a does not attack b, then there exists no $h \in \texttt{Supp}(b)$ such that $\{\texttt{Conc}(a), h\}$ is inconsistent. By maximality we conclude that $\texttt{Supp}(b) \subseteq A$. But according to the conclusion above, $\texttt{Supp}(b) \cup \{h\}$ is inconsistent which is a contradiction with the fact that A is consistent. ■

As a result of this proposition we get the following.

Proposition 3.3.9. Let $a \in \mathcal{A}$ be an argument of the form $a = (A, \bigwedge A)$ such that A is a maximal consistent set of facts. Then, for all argument b such that $\texttt{Supp}(b) \subseteq \texttt{Supp}(a)$ then a defends b. Consequently, $\{b, a\}$ is admissible.

Proof (Direct)**.** *If there exists an argument c such that c attacks b then there exists $h \in \texttt{Supp}(b)$ such that $\{\texttt{Conc}(c), h\}$ is inconsistent, and since $h \in \texttt{Supp}(a)$ then c attacks also b. From Proposition 3.3.8 the argument a defends itself from all attacks, hence a attacks c.* ■

We have seen in Proposition 3.3.6 that argumentation frameworks over **consistent** knowledge bases have always one non-empty extension under the preferred semantics, thus non-trivial. In what follows we prove that when $\mathcal{K}$ is inconsistent (i.e. $\mathcal{H} \in \textsc{arg}_\exists$) then $\mathcal{H}$ stays non-trivial.

Proposition 3.3.10 (Non-triviality). $\forall \mathcal{H} \in \textsc{arg}_\exists$, $\texttt{Ext}_p(\mathcal{H}) \neq \emptyset$.

Proof (Contradiction). *Suppose that* $\exists \mathcal{H} \in \textsc{arg}_\exists$ *such that* $\texttt{Ext}_p(\mathcal{H}) = \{\{\}\}$. *This means that there exists only the empty set as the admissible set of* $\mathcal{H}$. *This is in contradiction with Proposition 3.3.8 which states that the argument* $a = (A, \bigwedge A)$ *such that* A *is a maximal consistent set of facts is admissible. Thus, there exists a non-empty admissible set.* ■

From the proof of this proposition we can affirm easily that $\textsc{arg}_\exists$ contains no argumentation framework with rejected arguments.

Proposition 3.3.11 (Rejected arguments). For all $\mathcal{H} \in \textsc{arg}_\exists$:

$$\mathcal{A}_\mathcal{H} = \bigcup_{\mathcal{E} \in \texttt{Ext}_x(\mathcal{H})} \mathcal{E}$$

Such that $x \in \{p, s\}$.

Proof . *According to Proposition 3.3.9 for every argument* b *there exists an argument* $a = (A, \bigwedge A)$ *such that* $\texttt{Supp}(b) \subseteq \texttt{Supp}(a)$ *and* A *is a maximal consistent of facts where* $\{b, a\}$ *is admissible. Therefore all arguments are defended, hence there are no rejected arguments.* ■

It means that all the arguments are credulously accepted under the preferred/stable semantics.

Coherent argumentation frameworks (denoted $\textsc{coherent}$) are very interesting class of argumentation frameworks. This class received a particular interest in the community where many algorithms and proof procedures have been developed for such class (Modgil and Caminada (2009b)). In what follows we prove that $\textsc{arg}_\exists$ is within the class of coherent argumentation frameworks. Note that this has been proven in Croitoru and Vesic (2013).

Proposition 3.3.12 (Coherence). $\textsc{arg}_\exists \subseteq \textsc{coherent}$.

The coherence is very interesting as it reduces the complexity of the skeptical preferred acceptance of arguments to the case of skeptical stable which is less demanding, i.e. from Π_2^p-c to co-np-c. The next step is to see whether the skeptical preferred/stable for $\textsc{arg}_\exists$ coincides with the grounded acceptance. It has been proven in Croitoru and Vesic (2013) that if an argumentation framework has no rejected arguments under the preferred semantics then it is relatively grounded. From Proposition 3.3.11 we conclude the following:

Proposition 3.3.13 (Relative groundedness). $\textsc{arg}_\exists \subset \textsc{rgrounded}$.

This makes the complexity of skeptical preferred/stable decreases to the complexity of acceptance under grounded semantics which is in $\textsc{p}$.[7]

The proper inclusion is due to the fact that $\textsc{arg}_\exists$ are not well-founded. Recall that an argumentation framework is well-founded if and only if it is acyclic (has no cycle); Doutre (2002). To prove that we need to show that argumentation frameworks of $\textsc{arg}_\exists$ are cyclic (contain always cycles).

Lemma 3.3.1. Let $\mathcal{K}$ be the knowledge base of an argumentation framework $\mathcal{H} \in \textsc{arg}_\exists$. For all minimal conflicts $C = S \cup S'$ of $\mathcal{K}$, the attack between $a = (S, \bigwedge S)$ and $b = (S', \bigwedge S')$ such that S and S' are non-empty sets of facts and $|S| = |S'| = |C| - 1$ is symmetric.

Proof . *We show that the two arguments are valid and a attacks b and b attacks a.*

- *It is clear that* $\texttt{Supp}(a)$ *and* $\texttt{Supp}(b)$ *are consistent because by definition any subset of a minimal conflict is consistent (Definition 3.2.14, page 47).*

- *Let us prove that a attacks b. By definition there exists $h \in S'$ such that $C = S \cup \{h\}$ (it complements S to form C). Consequently, $S \cup \{h\}$ is inconsistent, in other words $\{\texttt{Conc}(a) \cup h\}$ inconsistent, that means a attacks b.*

- *Let us prove that b attacks a. By definition there exists $h \in S$ such that $C = S' \cup \{h\}$. Consequently, $S' \cup \{h\}$ is inconsistent, in other words $\{\texttt{Conc}(b) \cup h\}$ is inconsistent, that means b attacks a.*

■

Proposition 3.3.14. $\forall \mathcal{H} \in \textsc{arg}_\exists, \mathcal{H}$ are always cyclic.

Proof . *It follows from Lemma 3.3.1 in the sense that for all $\mathcal{H} \in \textsc{arg}_\exists$ there exists always a symmetric attack in $\mathcal{X}$, consequently a cycle, therefore $\mathcal{H}$ is cyclic.* ■

The following proposition is immediate.

Proposition 3.3.15 (Well-foundedness). $\textsc{arg}_\exists \cap \textsc{wfounded} = \emptyset$.

Figure 3.4 shows the relation between the class $\textsc{arg}_\exists$ and well-founded, trivial and coherent classes.

[7] For complexity results see Chapter 2, Table 2.2, page 25.

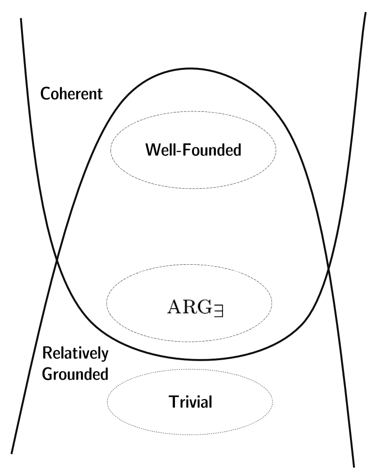


Figure 3.4: Relation between ARG$_\exists$ and other classes.

3.3.5 Postulates satisfaction

In this subsection we prove that ARG$_\exists$ comprises only argumentation frameworks that satisfy the rationality postulates. These postulates ensure that the output of the argumentation framework does not produce inconsistencies. In Caminada and Amgoud (2007) a set of postulates have been proposed for rule-based argumentation frameworks (i.e. frameworks that use strict and defeasible rules). Then, in Amgoud (2014) a more general set of postulates have been proposed for argumentation frameworks that are grounded on classical logics (Tarski's logics). In this thesis we are concerned with the latter.

In what follows we recall the postulates presented in the beginning of the section. Next, every postulate will be formally defined when we attempt to prove it.

(`Rationality postulates` Amgoud (2014))

- (P1) *Closure under $C\ell$*: the set of conclusions that can be drawn from any extension should be closed under $C\ell$.
- (P2) *Closure under sub-arguments*: if an argument is accepted in an extension then so are all its sub-arguments.
- (P3) *Consistency*: the set of conclusions that can be drawn from any extension should be consistent.
- (P4) *Exhaustiveness*: if an argument is accepted in an extension, then all its sub-parts should also be accepted in that extension
- (P5) *Free precedence*: any argument that is built only from the inconsistency-free part of the knowledge base it should be in every extension.

In Croitoru and Vesic (2013) direct consistency, indirect consistency and closure of Caminada and Amgoud (2007, 2005) has been proven. In Amgoud (2014), it has been shown that direct consistency and indirect consistency coincide giving rise to Postulate P3. We start by proving P2 which will provide with P3 the result that $\text{ARG}_\exists$ satisfies *strong consistency*. Finally, we prove P4 and P5. Since the preferred and stable coincide for $\text{ARG}_\exists$, throughout this section the set of extensions $\texttt{Ext}(\mathcal{H})$ refers to the one under the preferred semantics.

It has been shown in (Amgoud, 2014, Proposition 29) that argumentation frameworks whose attack relation satisfies the following condition are closed under sub-arguments:

- $\forall a, b, c \in \mathcal{A}$ such that $\texttt{Supp}(a) \subseteq \texttt{Supp}(b)$ if c attacks a then c attacks b.

Proposition 3.3.16. $\forall \mathcal{H} \in \text{ARG}_\exists$ the attack relation satisfies the condition above.

Proof . *If c attacks a then there exists $h \in \texttt{Supp}(a)$ such that $\{\texttt{Conc}(c), h\}$ is inconsistent. Since $\texttt{Supp}(a) \subseteq \texttt{Supp}(b)$ then $h \in \texttt{Supp}(b)$ therefore c attacks b.* ■

Corollary 3.3.2 (Closure under sub-arguments). $\forall \mathcal{H} \in \text{ARG}_\exists, \forall \mathcal{E} \in \texttt{Ext}(\mathcal{H})$: if $a \in \mathcal{E}$ then $\forall a' \in \mathcal{A}, a' \in \mathcal{E}$ where a' is a sub-argument of a.

Proof . *Corollary of Proposition 3.3.16.* ■

The argumentation frameworks of $\textsc{arg}_\exists$ satisfy strong consistency. Which dictates that for every extension $\mathcal{E}$, $\texttt{Base}(\mathcal{E})$ is consistent. This property has not been proven in Croitoru and Vesic (2013).

Corollary 3.3.3 (Strong consistency). $\forall \mathcal{H} \in \textsc{arg}_\exists$, $\forall \mathcal{E} \in \texttt{Ext}(\mathcal{H})$, $\texttt{Base}(\mathcal{E})$ is consistent.

Proof . *Corollary of P2, P3 and (Amgoud, 2014, Proposition 10) that dictates that if an argumentation framework satisfies consistency and closure under sub-arguments, then it also satisfies strong consistency.* ■

We provide a proof for P4 (Exhaustiveness).[8]

Proposition 3.3.17 (Exhaustiveness). $\forall \mathcal{H} \in \textsc{arg}_\exists, \forall \mathcal{E} \in \texttt{Ext}(\mathcal{H}), \forall a = (H, C) \in \mathcal{A}$, if $H \cup \{C\} \subseteq \texttt{Concs}(\mathcal{E})$, then $a \in \mathcal{E}$.

Proof . *Assume that $\exists a = (H, C) \in \mathcal{A}$ such that $H \cup \{C\} \subseteq \texttt{Concs}(\mathcal{E})$ and $a \notin \mathcal{E}$. This means that there exists an argument $b \in \mathcal{E}$ such that b attacks a because $\mathcal{E}$ is a stable extension. Consequent, $\exists h \in \texttt{Supp}(a)$ such that $\{h, \texttt{Conc}(b)\}$ is inconsistent. However, we know that $\texttt{Supp}(a) \subseteq \texttt{Concs}(\mathcal{E})$ and $\texttt{Conc}(b) \in \texttt{Concs}(\mathcal{E})$. This indicates that $\texttt{Concs}(\mathcal{E})$ is inconsistent which is in contradiction with the Consistency postulate.* ■

We finish the section by proving P5 (Free Precedence). To achieve that we prove that the attack relation of $\mathcal{H}$ is conflict-dependent.

Lemma 3.3.2 (Conflict-dependent). $\forall \mathcal{H} \in \textsc{arg}_\exists$, $\forall a, b \in \mathcal{A}$, if $(a, b) \in \mathcal{X}$ then $\texttt{Supp}(a) \cup \texttt{Supp}(b)$ is inconsistent.

Proof . *If a attacks b then there exists $h \in \texttt{Supp}(b)$ such that $\{h, \texttt{Conc}(a)\}$ is inconsistent. Consequently, $h \cup C\ell^*_{\mathcal{R}}(\texttt{Supp}(a))$ is inconsistent, therefore $\texttt{Supp}(b) \cup C\ell^*_{\mathcal{R}}(\texttt{Supp}(a))$ is inconsistent, hence $\texttt{Supp}(a) \cup \texttt{Supp}(b)$ is inconsistent.*

■

The following proposition is from (Amgoud, 2014, Proposition 36).

Proposition 3.3.18. For all argumentation systems $\mathcal{H} = (\mathcal{A}, \mathcal{X})$ such that $\mathcal{X}$ is conflict-dependent, $\mathcal{H}$ satisfies free precedence under grounded, ideal, complete, semi-stable and preferred semantics.

[8] This proof has been written independently from the one of (Amgoud, 2014, Proof of Proposition 14), page 36.

This yields the following proposition.

Proposition 3.3.19 (Free precedence). $\forall \mathcal{H} \in \textsc{arg}_\exists, \forall \mathcal{E} \in \texttt{Ext}(\mathcal{H})$:

$$\texttt{Args}(\texttt{Free}(\mathcal{K})) \subseteq \mathcal{E}.$$

Where $\mathcal{K}$ is the argumentation framework over which $\mathcal{H}$ has been built and $\texttt{Free}(\mathcal{K})$ is $\mathcal{F} \setminus C$ for all $C \in \texttt{conflicts}(\mathcal{K})$ and $\texttt{conflicts}(\mathcal{K})$ is the set of all minimal conflicts of $\mathcal{K}$.

In the next section we show the relation between the output of argumentation frameworks in $\textsc{arg}_\exists$ and CQA semantics. The goal is to show how argumentation can explain answers under the CQA semantics.

3.3.6 Relation with consistent query answering

Given an inconsistent knowledge base $\mathcal{K} = (\mathcal{F}, \mathcal{R}, \mathcal{N})$, a repair is a maximal (w.r.t $\subseteq$) consistent set of facts $F \subseteq \mathcal{F}$. We present the results of Croitoru and Vesic (2013) that show that there is a correspondence between repairs $\mathcal{R}epair(\mathcal{K})$ and the extensions $\texttt{Ext}(\mathcal{H}_\mathcal{K})$ of argumentation frameworks of $\textsc{arg}_\exists$ under the preferred (and stable) semantics.

Theorem 3.3.1 (Croitoru and Vesic (2013)). Let $\mathcal{K}$ be a knowledge base, $\mathcal{H}_\mathcal{K} \in \textsc{arg}_\exists$ its corresponding argumentation framework. Then:

$$\texttt{Ext}_p(\mathcal{H}_\mathcal{K}) = \{\texttt{Args}(\mathcal{P}) \mid \mathcal{P} \in \mathcal{R}epair(\mathcal{K})\}$$

After establishing the correspondence between the extension of preferred semantics with repairs, let us show the equivalence between reasoning under argumentation semantics and CQA semantics.

Theorem 3.3.2 (Croitoru and Vesic (2013)). Let $\mathcal{K} = (\mathcal{F}, \mathcal{R}, \mathcal{N})$ be a knowledge base, let $\mathcal{H}_\mathcal{K}$ be the corresponding argumentation framework and let $\mathcal{Q}$ be a query. Then:

- $\mathcal{K} \models_{\mathcal{CQA}} \mathcal{Q}$ iff $\forall \mathcal{E} \in \texttt{Ext}_p(\mathcal{H}_\mathcal{K}), \mathcal{E} \models \mathcal{Q}$.

Example 3.3.10. *Consider the knowledge base of Example 3.3.6. The repairs $\mathcal{R}epair(\mathcal{K})$ are:*

- $\mathcal{P}_1 = \{jaguar(T), felidae(T)\}$
- $\mathcal{P}_2 = \{leopard(T), felidae(T)\}$

The extensions are:

- $\mathcal{E}_1 = \{a_1, a_4, a_5, a_7, a_8, a_9, a_{10}, a_{11}, a_{17}\}$.
- $\mathcal{E}_2 = \{a_2, a_3, a_6, a_{12}, a_{13}, a_{14}, a_{15}, a_{16}, a_{17}\}$.

We have the following equivalences:

- $\mathcal{Q} = felidae(T)$ *is accepted under CQA semantics.*
- $\mathcal{Q} = felidae(T)$ *is universally accepted.*
- $\mathcal{Q}' = animal(T)$ *is accepted under CQA semantics but not skeptically accepted. But it is universally accepted.*
- $\mathcal{Q}'' = jaguar(T)$ *is not accepted under CQA semantics and not universally accepted.*

An important corollary of Theorem 3.3.2 is that CQA semantics and universal acceptance are equivalent.

Corollary 3.3.4. A query $\mathcal{Q}$ is accepted under CQA semantics if and only if it has a proponent set.

This gives us the possibility to use One-shot Argument-based Explanations (blocks and proponent sets) of Subsection 3.3.3 to explain CQA answering in existential rules. For instance to explain why $\mathcal{Q}'$ is accepted under CQA we use the proponent set "T is an animal because he is either a jaguar or a leopard". To explain why $\mathcal{Q}''$ is not accepted under CQA we use the block "T is not a jaguar because it is possible that he is a leopard".

3.4 Conclusion

In this chapter we have introduced argumentation theory in its abstract and logical form. For the latter, we have considered the logic-based instantiation of Dung's abstract argumentation of Croitoru and Vesic (2013) that is grounded on the existential rules framework.

For such instantiation, we have shown that the concept of skeptical acceptance of arguments does not necessarily yield all plausible conclusions. In fact there are other conclusions which are plausible (belong to the output) but their supporting arguments are not in all extensions. We proved that the set of plausible conclusions contains skeptical conclusions that follow from skeptical arguments and universal conclusions that are not necessarily conclusions of skeptically accepted arguments (Proposition 3.3.2). As the characterization of skeptical acceptance is well-studied in the community, we turned to universal and non-universal acceptance for which we have given a precise characterization. The main results are :

- Proposition 3.3.4: a conclusion is universally accepted iff it has a proponent set. A proponent set is a minimal set of supporting arguments that contains an argument from each extension.
- Proposition 3.3.5: a conclusion is not universally accepted iff it has a block. A block is a set of arguments that can be extended to form an extension and attacks all the supporters of the conclusion.
- The two problems are defined in terms of the hitting set problem.
- Blocks and proponent sets are in fact explanations for non-universal acceptance and universal acceptance respectively. Since these explanations are not interactive (not dialogical) they are referred to as *One-shot Argument-based Explanations*.

After studying the properties of the output, we shifted in Subsection 3.3.4 to the study of the properties of the class of such argumentation frameworks, i.e. the class $\textsc{arg}_\exists$. The main results that we have shown is that every argumentation framework $\mathcal{H} \in \textsc{arg}_\exists$ is:

- Finite: arguments are finite and the set $\mathcal{A}_\mathcal{H}$ is finite and the attack relation is finitary.
- Coherent: the stable, preferred extensions coincide. As a trivial consequence, semi-stable coincides too with stable and preferred.
- Not well-founded: the argumentation framework is not well-founded, therefore the argument graph is not acyclic.
- Relatively grounded: the grounded extension coincides with the intersection of all preferred extensions.
- Non-trivial: the empty extension is never the only preferred extension of $\mathcal{H}$.

Next, we have proven that the instantiation does respect the *augmented* rationality postulates of Amgoud (2014). Namely, closure under subarguments, consistency, exhaustiveness and free precedence. Then we showed that these postulates are not respected under the naive semantics. However, the satisfaction of these postulates under the preferred semantics naturally yields[9] to a full correspondence between the preferred extensions of $\mathcal{H}_{\mathcal{K}}$ and repairs of $\mathcal{K}$. In Subsection 3.3.6 we explored such correspondence which has already been proven by Croitoru and Vesic (2013), this relation is in the heart of our thesis because the correspondence between universal acceptance and CQA semantics allows us to use the explanatory power of argumentation to explain CQA answers using *One-shot Argument-based Explanations.* However, this type of explanation lacks interactiveness with the user and its computation is based on the computation of extensions (which is computationally hard). In the next section we use another approach for explanation which is dialogical (i.e. interactive) and does not necessitate the computation of extensions. We introduce Meta-level Dialectical Explanations which are based on the concept of dialectical proof theories in argumentation.

[9] According to Croitoru and Vesic (2013); Amgoud and Besnard (2013); Vesic (2013).

4 Meta-level Dialectical Explanations

In this chapter we are interested in the problem of universal and non-universal acceptance in logic-based argumentation frameworks under the preferred/stable semantics. We show in this chapter how this problem can be solved through a dialogue game between an opponent and a proponent over a given query. This dialogue results in what is called *a dialectical proof* which is the meta-level dialectical explanation for the query in question. The goal of proposing a dialectical proof theory is twofold: (1) provide a computational procedure that decides query acceptance without the need to compute extensions, and most importantly (2) provide a formal ground for interactive explanations that overcome the limitations of one-shot argument-based explanations. We start in Section 4.1 by an introduction and an informal description of dialectical proof theories for abstract argumentation with a discussion on the existing dialectical proof theories. We show through a critical example why the state-of-the-art dialectical proof theories cannot be applied to our context. Then, in Section 4.2 we propose a dialectical proof theory for universal and non-universal acceptance under the preferred/stable semantics. Then, in Section 4.3 we explain how the proof theory applies on a detailed example. In Section 4.4 we prove the soundness and completeness of the theory and we study the dispute complexity of dialectical proofs alongside other properties. Finally, in Section 4.5 we empirically evaluate the effect of meta-level dialectical explanations on users with respect to different criteria. We report how they impact the accuracy of users when faced with inconsistent situations. Moreover, we investigate how the users find meta-level dialectical explanations with respect to clarity and intelligibility.

4.1 Introduction

Dialectical proof theories have their roots in the dialogical approach to logic (Lorenz, 2001). In the Greek antiquity, logic was studied in a dialogical context where two parties exchange arguments over a central claim. In modern logic, the dialogical approach or *dialogical logics* is used to provide a game-theoretical semantics for logical systems. Precisely, they provide a constructive proof for the notion of validity in classical or non-classical logics. This proof is presented as a dialogue game between two parties arguing about a thesis while respecting some fixed rules. The dialogue is adversarial where one party plays the role of the defender of the thesis (proponent) and the other argues against the thesis (opponent). Any dialogue ends after a finite number of moves with a winner and a loser.

After the introduction of abstract argumentation framework in Dung (1995), many attempts have been made to adapt this dialogical approach to provide formal proof theories for abstract argumentation. It is often referred to as *dialectical proof theories* where the adjective "dialectical" is due to the conversational aspect of the proof. Jakobovits and Vermeir (1999); Prakken (2001) define, similarly to dialogical logic, a dialectical proof theory as an argument game with a winning criterion alongside with a legal move function that decides the allowed moves to be played. Given an argumentation framework, a semantics x and an argument a, the goal is to prove whether the argument a is skeptically/credulously accepted under the semantics x. We say a dialectical proof theory is sound if it proves only skeptically/credulously accepted arguments (it does not prove what is false). We say it is complete if it can prove every skeptically/credulously accepted arguments (it proves all what is true).

The TPI (Two Party Immediate Response) procedure proposed in Vreeswik and Prakken (2000) and further formalized in Dunne and Bench-Capon (2003) is used for credulous and skeptical games in finite and coherent argumentation frameworks where two players exchange arguments (moves) until one of them cannot play. The justification state of the argument (skeptical/credulous) is decided with respect to the wining criterion. The turn in TPI-disputes shifts after one move with the move m_i attacks the precedent one (hence *immediate response*). Their dialectical proof theories are sound and complete. In Cayrol et al. (2003), the same guideline is followed but with a refinement on the size of the proof, where Cayrol et al. (2003) produce shorter proofs than Dunne and Bench-Capon (2003). In Modgil and Caminada (2009b) a different dialectical proof theory has been proposed for skeptical acceptance where, instead of exchanging arguments the proponent and the opponent exchange whole admissible sets. The goal is to construct a *block*, which is an admissible set of arguments that conflicts with all admissible sets around the argument in question (Modgil and Caminada, 2009b, Theorem 6.7). Following the same idea, Doutre and Mengin (2004) construct such block in a *meta-argumentation framework* within a *meta-dialogue* where admissible sets are considered as moves, then the classical credulous proof theory of Cayrol et al. (2003) is used as a sub-procedure to proof skeptical acceptance. In Thang et al. (2009) a more general framework has been provided which is sound for any argumentation framework and it is complete for general classes of finitary argumentation frameworks including the class of finite argumentation frameworks using the notions of dispute derivation and base derivation. For skeptical preferred, the proof theory proposes to find a base then check whether it is complete or not. A base of a is a set of admissible sets that contain a such that whenever a is in an extension then there is an admissible set in the base that belongs to this extension. The base is complete if all extensions contain an admissible set from the base.

When it comes to logic-based argumentation, the situation is quite different. In logic-based argumentation we differentiate between the acceptance of an argument and the acceptance of a conclusion. As mentioned in Chapter 3, Example 3.3.5,, page 56 the universal acceptance of a conclusion does not necessarily entail the skeptical acceptance of its argument(s), whereas the skeptical acceptance of an argument necessarily entails the skeptical acceptance of its conclusion (consequently the universal acceptance). For this reason, applying the already proposed dialectical proof theories will fail simply because they handle a different problem. In the next section, following Dunne and Bench-Capon (2003) we propose a new TPI-like dialectical proof theory for universal acceptance.

4.2 Universal Acceptance Dialectical Proofs

Given a query $\mathcal{Q}$ and an argumentation framework $\mathcal{H}$, the preferred universal dialectical proof theory is a two-person argument game between a proponent (`PRO`) and an opponent (`OPP`). `PRO` takes the position of supporting the query $\mathcal{Q}$ while `OPP` takes the opposite. The proponent and the opponent are engaged in an argumentation dialogue of *precisely defined* types of moves. The goal is to determine at the end of the dialogue whether the query is universally accepted or not. If the query is universally accepted (or not) the dialogue is considered as a dialectical proof for its justification state.

Let us formally define what is a dialogue in this proof theory.

Definition 4.2.1 (Dialogue). *Let $\mathcal{H} = (\mathcal{A}, \mathcal{X})$ be an argumentation framework. A dialogue based on $\mathcal{H}$ is a finite sequence $d_n = (m_1, \ldots, m_n)$ of moves where each m_j is either:*

- ***Support move:*** $m_j = \textsc{support}(a)$ *such that* $a \in \mathcal{A}$ *(In this case we denote* $\mathtt{Arg}(m_j) = a$ *and* $\mathtt{Sp}(m_j) = \textsc{support}$*).*
- ***Counter move:*** $m_j = \textsc{counter}(A)$ *such that* $A \subseteq \mathcal{A}$ *(In this case we denote* $\mathtt{Arg}(m_j) = A$ *and* $\mathtt{Sp}(m_j) = \textsc{counter}$*).*
- ***Retrace move:*** $m_j = \textsc{retrace}(A, i)$ *such that* $A \subseteq \mathcal{A}$ *and* $i < j$ *(In this case we denote* $\mathtt{Arg}(m_j) = A$*,* $\mathtt{Sp}(m_j) = \textsc{retrace}$*).*

Odd-indexed (resp. even-indexed) moves are played by PRO *(resp.* OPP*). We denote by* $d \cdot d'$ *and* $d \cdot m$ *the concatenation of the dialogues* d *and* d' *and the dialogue* d *with the move* m *respectively. The retrace move has a special parameter* i *called the index (denoted as* $\mathtt{Idx}(m)$*). The subscript of* d_n *refers to the stage of the dialogue. We may abuse notation and we write* $m_i \in d_n$ *to mean that* m_i *is in* d_n*.*[1]

A dialogue is a sequence of moves with different types of moves respecting a turn taking mechanism. The turn taking mechanism is simple and deterministic where odd indexed moves are advanced by PRO while even index moves are advanced by OPP. The moves of the dialogue are defined in terms of speech acts and content which can express, support, counter attack or retrace. The move $\textsc{support}(a)$ advances an argument a which supports an arbitrary query. The move $\textsc{counter}(A)$ counterattacks the position of PRO by advancing a set of arguments. The move $\textsc{retrace}(A, i)$ is used to retrace to earlier stage in the dialogue and continue from thereafter. The first move can only be played by PRO, whereas COUNTER and RETRACE are only employed by OPP.

In this dialectical theory, any dialogue starts by PRO advancing a support move to support the query in question. Then, OPP presents an argument (or a set) that attacks the previously advanced argument. Next, PRO try to avoid this attack and reinstate the query using another argument which is not attacked by the already advanced attackers. OPP in turn, tries to extend the previous set of attackers so that it attacks all the supporters advanced so far. When OPP fails to extend the set, he retraces back and chooses another set of arguments and continues the dialogue from thereafter. By doing so OPP is somehow trying to construct a set of arguments that attacks all the supporters of the query $\mathcal{Q}$. In other words, he is trying to build a *block* for the query $\mathcal{Q}$ (cf. Definition 3.3.13, page 61).

Many questions can rise, for instance, what happens when OPP retraces? would PRO play the supporters which were attacked before retracing or not?

[1] We may sometimes omit the subscript when it is not needed.

what is the nature of the advanced sets of arguments in COUNTER?

In order to answer these question (and others) we need to introduce a control structure that keeps track of the state of the dialogue. This structure will be used in defining the nature and legality of moves.

Definition 4.2.2 (Dialectical state)**.** *Let d_k be a dialogue at stage k. The dialectical state of d_k is a tuple $\delta_k = (\pi_k, h_k, \theta_k, \beta_k, \Delta_k)$*[2]*:*

- *π_k: the set of arguments available to* `PRO`*.*
- *h_k: the set of arguments that have been played so far by* `PRO`*.*
- *θ_k: the set of arguments available to* `OPP`*.*
- *β_k: the current block constructed by* `OPP`*.*
- *Δ_k: the sets of arguments that have been shown to be not blocks.*

d_0 is the empty dialogue and δ_0 is its initial dialectical state.

This state defines at any stage k of the dialogue d_k the set of arguments π_k available to `PRO` that can be used to support the query $\mathcal{Q}$. In the dialectical state, we find also the set h_k which shows the arguments that have been so far played by `PRO`. In addition, it presents the set θ_k of arguments that can be used to attack the arguments previously advanced by `PRO`. β_k presents the currently constructed block. When `OPP` fails to extend the current block to another that attacks all the previously played supporters, he uses the RETRACE move. By doing so we need to keep track of the sets of arguments that cannot be extended to a block. These are stored in Δ_k.

Given a query $\mathcal{Q}$, the initial state of the dialectical state is described as follows:

- $\pi_0 = \mathtt{S}(\mathcal{Q})$.
- $h_0 = \emptyset$, $\theta_0 = \emptyset$, $\beta_0 = \emptyset$, $\Delta_0 = \emptyset$.

Since the dialogue d_0 has not yet been started, the set of available arguments π_0 for `PRO` ranges over all the possible supporters of the query $\mathcal{Q}$. The played arguments h_0, the available arguments θ_0, current block β_0 and Δ_0 are empty since the first move has not yet been uttered.

[2]To be able to understand the terms think of π as the first letter of **p**roponent, h as **h**istory, θ as **o**pponent and β as **b**lock.

The advancement of moves within the dialogue are usually controlled by a legal move function (Modgil and Caminada, 2009b) which can be expressed in terms of rules, called dialogue rules. Every move depends on certain *preconditions* about the actual dialectical state and the previous move advanced by the other party. Every move also determines the possible moves to be played next (*postcondition rules*).

Moves affect the dialectical states of the dialogue. In fact, we can see the moves as transitions between possible dialectical states where a dialectical state δ_k for a dialogue d_k and a move m_{k+1} define a new dialectical state δ_{k+1}. This is called the *effect* of the move.

In what follows we recall for each move its informal description and we present the preconditions that should be satisfied so that the move is considered legal to be played. We present then its effect on the dialectical state and its postconditions.

Let d_k be a dialogue and δ_k the current dialectical state of d_k. Let m_{k+1} be a move and δ_{k+1} be the dialectical state of the dialogue $d_{k+1} = d_k \cdot m_{k+1}$ after playing the move m_{k+1}. Note that for a given move we index preconditions (resp. effects) by the first letter of the speech act of the move followed by P (resp. E) and subscripted by a number.

Move:

$m_{k+1} = \textsc{support}(a)$.

Description:

this move advances an argument that supports the query in question.

Preconditions:

(SP$_1$) $k + 1$ is odd.

(SP$_2$) $a \in \pi_k$.

Postconditions rules:

the next move can be either COUNTER or RETRACE.

Effects:

(SE$_1$) $\pi_{k+1} = \pi_k / a$.

(SE$_2$) $h_{k+1} = h_k \cup \{a\}$.

(SE$_3$) $\theta_{k+1} = \texttt{range}^-(h_{k+1})$.

(SE$_4$) $\beta_{k+1} = \beta_k$.

(SE$_5$) $\Delta_{k+1} = \Delta_k$.

This move is advanced by PRO, therefore $k+1$ should be odd (SP_1). It advances an argument a that supports the query $\mathcal{Q}$ which is not attacked by the current block β_k presented so far (SP_2). To respond to this move, in the next turn OPP should either use COUNTER or RETRACE.

As one may notice, the support move m_{k+1} changes the set of available arguments π_{k+1} of PRO. In fact a supporting argument seizes to be available once it is played (SE_1). In contrast it is added to the history h_{k+1}. The support move alters the set of available arguments of OPP by adding all arguments that can be played in the future by OPP (SE_3).

As indicated in the postconditions of the support move, a counter move is allowed to be played next.

Move:

$m_{k+1} = \text{COUNTER}(A)$.

Description:

this move advances a set of arguments that attacks all the arguments presented so far.

Preconditions:

(CP_1) $k+1$ is even.

(CP_2) $A = \beta_k \cup S$ such that $S \subseteq \theta_k$ (i.e. A extends β_k by S).

(CP_3) A attacks h_k and belongs to (or is) an admissible set.

(CP_4) there is no $A' \in \Delta_k$ such that $A' \subseteq A$.

Postconditions rules:

the next move should be SUPPORT.

Effect:

(CE_1) $\pi_{k+1} = \pi_k/\mathtt{range}^+(A)$.

(CE_2) $h_{k+1} = h_k$.

(CE_3) $\theta_{k+1} = \theta_k$.

(CE_4) $\beta_{k+1} = A$.

(CE_5) $\Delta_{k+1} = \Delta_k$.

This move is advanced by OPP therefore $k+1$ should be even (CP_1). It tries to extend the current block β_k to another set of arguments that attacks all the supporters presented so far (CP_2 and CP_3). OPP does so by incorporating arguments from θ_k. The new current block ($\beta_{k+1} = A$) or one of its subsets should have not been already proven to be not a block (CP_4). After advancing m_{k+1}, π_{k+1} contains all the arguments from π_{k+1} except those which are attacked by A (CE_1), thus they are spared from further use. Note that the spared arguments may be radded afterwards, this is particularly the case when we use retrace as we shall mention later.

The set of available moves θ_{k+1} of `OPP` contains all the arguments that attack the supporting arguments that can be played by `PRO` (CE_3). Since A attacks all the supporting arguments so far provided, it is considered the current block (CE_4). The set Δ_{k+1} and h_{k+1} are left unchanged (CE_2 and CE_5).

After a support move, `OPP` can also play a retrace move. This is particularly needed when he is unable to play a counter move. The formal details about the retrace move are presented hereafter:

Move:

$m_{k+1} = \text{RETRACE}(A, i)$.

Description:

this move retraces to the recent stage i from which it can extend the current block of i.

Preconditions rules:

(RP_1) $k + 1$ is even, $i < k + 1$ and i is odd.

(RP_2) there is no set of arguments $S \subseteq \theta_k$ such that $\beta_k \cup S$ is (or belongs to) an admissible set and attacks h_k.

(RP_3) $A = \beta_i \cup S$ such that $S \subseteq \theta_i$.

(RP_4) A attacks h_i and belongs to (or is) an admissible set.

(RP_5) there is no $A' \in \Delta_k$ such that $A' \subseteq A$.

Postconditions:

the next move should be SUPPORT.

Effect:

(RE_1) $\pi_{k+1} = \pi_i/\texttt{range}^+(A)$.

(RE_2) $h_{k+1} = h_i$.

(RE_3) $\theta_{k+1} = \theta_i$.

(RE_4) $\beta_{k+1} = A$.

(RE_5) $\Delta_{k+1} = \Delta_k \cup \beta_k$.

When `OPP` cannot extend the current block β_k with arguments from θ_k (RP_2), he should retrace back and choose other arguments. The index i (which should be odd) determines the point of a support move from which `OPP` can mount another line of attack. By starting a new line of attack, `OPP` should opt for a new block that attacks all the supporters from Stage i up to the Stage 1 (RP_3) by extending β_i from θ_i. The new block $\beta_{k+1} = A$ or one of its subsets should have not been already proven to be not a block (CP_5).

When the retrace move is advanced, π_{k+1} is reset to its ancient state i in addition to excluding all the arguments that can be attacked afterwards (RE_1). The current block β_{k+1} is set to A (RE_4), while Δ_{k+1} is set to $\Delta_k \cup \beta_k$ (RE_5), i.e. the block of stage k which `OPP` could not extend.

If one of the preconditions is not satisfied, `OPP` go further and look for other stages where he can mount a new attack. `OPP` follows the following procedure:

Procedure 4.2.1. *Let d_n be a dialogue and m_n be the last played move such that $\mathtt{Sp}(m_n) = \textsc{support}$. If* `OPP` *cannot play a counter move m_{n+1} then it tries to play the retrace move m_{n+1} as follows:* [3]

1. *do $y = y - 1$ until $m_y = \textsc{retrace}(A, x)$ or $m_y = \textsc{support}(a)$ or $y = 0$.*

2. *if $m_y = \textsc{retrace}(A, x)$ then:*

   (a) *If there does not exist a move $m_{n+1} = \textsc{retrace}(A', x)$ that respects the preconditions then set $y = x$ and goto 1 else play m_{n+1} and exit.*

3. *if $m_y = \textsc{support}(a)$ then:*

   (b) *If there does not exist a move $m_{n+1} = \textsc{retrace}(A', y)$ that respects the preconditions then goto 1 else play m_{n+1} and exit.*

`OPP` starts by looking for the most recent retrace or support move (line 1). If a retrace move is found (line 2) then it tries to play a retrace to stage x that respects the preconditions (line a) by looking exhaustively for all possible sets A' that makes the move respect the preconditions. If he succeeds to play such move, the procedure exits. Otherwise it continues the search by setting y to x. If a support move m_y is found (line 3) then it plays

[3] Note that y is initialized to n and $x < y$, and a, A are arbitrary (set of) arguments respectively.

a retrace with index to y. Otherwise, it continues the search for other moves from which OPP can play.

To better illustrate the point let us apply the procedure on an example.

Example 4.2.1. *Consider the dialogue example in the table of Figure 4.1. Let us see how* OPP *has played the retrace move at stage (8). Note that this is just an illustrative example and does not correspond to a real dialogue.*

At stage (7), PRO *played the argument c.* OPP *tried to play a counter move but he failed to do so. Now,* OPP *will follow Procedure 4.2.1 to play a retrace move. At this moment $n = 7$, $y = n$.* OPP *gets into the loop at line (1). When $y = 6$,* OPP *encounters the retrace move $m_6 = \text{RETRACE}(\{p, r\}, 3)$, he tries to play a retrace move $m_8 = \text{RETRACE}(A', 3)$ but it seems that he couldn't play such move because there is no A' that makes the retrace move respect the preconditions.* OPP *continues this time from 2 (y is set to 3 in line (b) and it gets decreased at line (1)) where he skips the counter move and stops at stage (1) where a support move $m_1 = \text{SUPPORT}(a)$ is found. At this point,* OPP *plays the retrace move $m_8 = \text{RETRACE}(\{s\}, 1)$ which seems to respect the preconditions. Afterwards, the dialogue continues normally by* PRO *until it stops at stage (12) with* OPP *playing the last move.*

In fact, the dialogue represents a compact representation of a tree where retrace moves represent branching points. This tree is called the associated dialogue tree and it is defined as follows.

Definition 4.2.3 (Dialogue tree)**.** *Given a dialogue $d_n = (m_1, \dots, m_n)$, its dialogue tree is a labeled tree $\mathcal{T}(d_n) = (\mathcal{V}, \mathcal{D})$ such that $\mathcal{V}$ is a set of nodes and $\mathcal{D}$ is a binary relation over $\mathcal{V}$ defined as follows:*

- $\mathcal{V} = \{\texttt{Arg}(m_i) \mid m_i \in d_n\}$.
- $\mathcal{D} = \{(\texttt{Arg}(m_{i-1}), \texttt{Arg}(m_i)) \mid i \leqslant n \text{ and } m_i \neq \text{RETRACE}(A, j)\} \cup \{(\texttt{Arg}(m_j), \texttt{Arg}(m_n)) \mid i \leqslant n \text{ and } m_i = \text{RETRACE}(A, j)\}$.

$\texttt{Arg}(m_1)$ *is the root node of the tree. Note that $|\mathcal{T}(d_n)| = |\mathcal{V}|$ refers to the size of the tree which is equal to the number of its nodes.*

It is a tree where nodes are arguments or set of arguments played by both parties. Odd-level nodes are played by PRO and even-level nodes are played by OPP.

Fact 4.2.1. Le d_n be a dialogue, $\mathcal{T}(d_n)$ its associated dialogue tree and $\texttt{Pre}(\mathcal{T}(d_n))$ the pre-order traversal of $\mathcal{T}(d_n)$. The following hold:

i	Move
1	SUPPORT(a)
2	COUNTER($\{p\}$)
3	SUPPORT(b)
4	COUNTER($\{p, q\}$)
5	SUPPORT(c)
6	RETRACE($\{p, r\}, 3$)
7	SUPPORT(c)
8	RETRACE($\{s\}, 1$)
9	SUPPORT(d)
10	COUNTER($\{s, t\}$)
11	SUPPORT(e)
12	RETRACE($\{s, v\}, 9$)

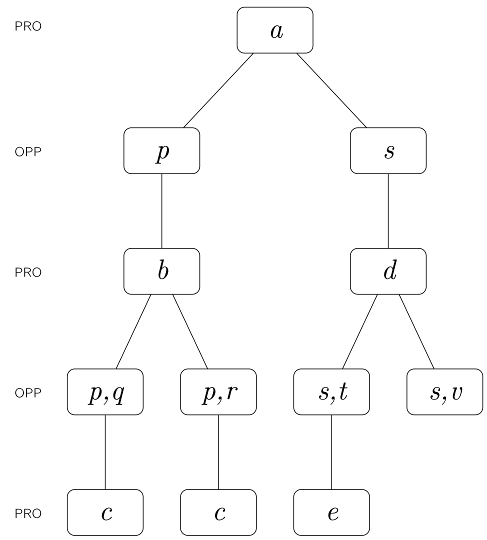


Figure 4.1: The left table presents the dialogue, the right figure shows its associated dialogue tree.

1. $\mathcal{T}(d_n)$ is unique.
2. $|d_n| = |\mathcal{T}(d_n)|$.
3. $\texttt{Pre}(\mathcal{T}(d_n)) = \texttt{Seq}(d_n)$.

Such that $\texttt{Seq}(d_n) = (c_1, \ldots, c_n)$ is the sequence of the content of moves played in d_n, i.e. $c_i = \texttt{Arg}(m_i)$.

Example 4.2.2. *The tree in Figure 4.1 represents the associated tree $\mathcal{T}(d_{12})$ of the dialogue d_{12} (shown in the table). The pre-order traversal of $\mathcal{T}(d_{12})$*

- $\texttt{Pre}(\mathcal{T}(d_{12})) = (a, \{p\}, b, \{p, q\}, c, \{p, r\}, c, \{s\}, d, \{s, t\}, e, \{s, v\})$

is the same as the sequence of content that can be extracted from d_{12}. Also, observe that $|\mathcal{T}(d_{12})| = 12$.

The dialogue terminates when no one can further the dialogue with moves.

Definition 4.2.4 (Termination and wining)**.** *Let d_k be a dialogue and δ_k the current dialectical state of d_k. The dialogue d_k is a terminated dialogue if and only if the player to play has run out of moves. The winner of the dialogue is* $\texttt{Player}(m_k)$ *(the player of the last move).*

It is easy to determine the winner of a dialogue from its tree.

Property 1. Le d_n be a terminated dialogue, $\mathcal{T}(d_n)$ its associated dialogue tree. The following statements are equivalent:

- The length of the right-most path is odd.
- PRO is the winner of d_n.

Proof (Sketch)**.** *It is clear that the last move is corresponds to the leaf node in the right-most path of the tree. If the length of the path is odd then* PRO *is the last one who played, consequently* PRO *is the winner.* ■

After having defined the theory, let us define the concept of a dialectical proof.

Definition 4.2.5 (Dialectical proof)**.** *Given a query $\mathcal{Q}$ and a terminated dialogue d_n. We call d_n a dialectical proof for the universal acceptance of $\mathcal{Q}$ if and only if* PRO *is the winner, otherwise it is called a dialectical proof for non universal acceptance of $\mathcal{Q}$.*

In the next section we give a detailed example of universal acceptance and non-universal acceptance on a real argumentation framework.

4.3 Example

Consider the argumentation framework $\mathcal{H}$ of Figure 4.2.(a). This argumentation framework is coherent. Suppose that the gray-colored arguments supports a query $\mathcal{Q}$ (i.e. $\texttt{S}(\mathcal{Q}) = \{a, d, e, l, h\}$). In what follows, we show how the query $\mathcal{Q}$ is universally accepted by providing a dialectical proof.

The dialectical proof is presented in Table 4.1 and its associated dialogue tree is shown in Figure 4.2.(b).

At stage (0) the dialectical state is initialized as defined previously. The dialogue starts at stage (1) by PRO playing the supporter a from the available supporters in π_0. When PRO plays a, the argument a is moved from the available supporters π_1 to the history of advanced arguments by PRO h_1. The set of available attackers θ_1 becomes the set of all attackers of $\pi_1 \cup h_1$.

i	Move	π_i	h_i	θ_i	β_i	Δ_i
0	-	$\{a, d, e, l, h\}$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$
1	$\text{S}(a)$	$\{d, e, l, h\}$	$\{a\}$	$\{g, i\}$	$\emptyset$	$\emptyset$
2	$\text{C}(\{g\})$	$\{e, l, h\}$	$\{a\}$	$\{g, i\}$	$\{g\}$	$\emptyset$
3	$\text{S}(l)$	$\{e, h\}$	$\{a, l\}$	$\{g, i, c, b\}$	$\{g\}$	$\emptyset$
4	$\text{C}(\{g, c\})$	$\{e, h\}$	$\{a, l\}$	$\{g, i, c, b\}$	$\{g, c\}$	$\emptyset$
5	$\text{S}(e)$	$\{h\}$	$\{a, l, e\}$	$\{g, i, c, b, k\}$	$\{g, c\}$	$\emptyset$
6	$\text{R}(\{i\}, 1)$	$\{d, e, l, h\}$	$\{a\}$	$\{g, i\}$	$\{i\}$	$\{\beta_5\}$
7	$\text{S}(d)$	$\{e, l, h\}$	$\{a, d\}$	$\{g, i, f, j, b\}$	$\{i\}$	$\{\beta_5\}$
8	$\text{C}(\{i, f\})$	$\{l, h\}$	$\{a, d\}$	$\{g, i, f, j, b\}$	$\{i, f\}$	$\{\beta_5\}$
9	$\text{S}(h)$	$\{l\}$	$\{a, d, h\}$	$\{g, i, f, j, b, k\}$	$\{i, f\}$	$\{\beta_5\}$
10	$\text{C}(\{i, f, k\})$	$\{l\}$	$\{a, d, h\}$	$\{g, i, f, j, b, k\}$	$\{i, f, k\}$	$\{\beta_5\}$
11	$\text{S}(l)$	$\emptyset$	$\{a, d, h, l\}$	$\{g, i, f, j, b, k, c\}$	$\{i, f, k\}$	$\{\beta_5\}$
12	$\text{R}(\{i, j\}, 7)$	$\{e, l, h\}$	$\{a, d\}$	$\{g, i, f, j, b\}$	$\{i, j\}$	$\{\beta_5, \beta_{11}\}$
13	$\text{S}(h)$	$\{e, l\}$	$\{a, d, h\}$	$\{g, i, f, j, b, k\}$	$\{i, j\}$	Δ_{12}
14	$\text{C}(\{i, j, k\})$	$\{l\}$	$\{a, d, h\}$	$\{g, i, f, j, b, k\}$	$\{i, j, k\}$	Δ_{12}
15	$\text{S}(l)$	$\emptyset$	$\{a, d, h, l\}$	$\{g, i, f, j, b, k, c\}$	$\{i, j, k\}$	$\Delta_{12} \cup \{\beta_{14}\}$

Table 4.1: A dialectical proof for the query $\mathcal{Q}$. For space reasons S(), C() and R() denote SUPPORT(), COUNTER() RETRACE() respectively.

This means when the turn of OPP comes at stage (2) he shall choose from this set. At stage (2) OPP advances a counter move with argument g that attacks all the advanced supporters (i.e. $h_1 = \{a\}$). After advancing such move, the argument d is removed from the set of available arguments π_2 since g attacks d, thus PRO will not be able to play d. Observe that j is removed also from θ_2 because it does not attack any argument in $\pi_2 \cup h_2$. Since $\{g\}$ attacks all the supporters advanced so far, it becomes the current block, i.e. $\beta_2 = \{g\}$. At stage (3), PRO responds by a support move with the argument l that is not attacked by the current block. At stage (4), OPP extends the current block $\beta_3 = \{g\}$ by the argument c which attacks l. Note that $\{g, c\}$ is a subset of the admissible set $\{g, c, e\}$. Now, $\beta_4 = \{g, c\}$ attacks all the presented supporters. At stage (5), PRO presents another unattacked supporter (i.e. e). Note that the choice of the supporters is arbitrary.

At stage (6), OPP could not extend the current block β_5 into another that attacks e too. Therefore OPP plays a retrace move $\text{R}(\{i\}, 1)$ that can be read as "retrace to stage (1) and play a counter move with $\{i\}$". By doing so, OPP creates another line of dialogue and rolls back all the changes that have been made on the dialectical state up to the stage (1). That is why at stage (6) the sets π_6, h_6 and θ_6 are set to π_1, h_1 and θ_1 respectively. The current block is changed to $\{i\}$ and the ancient block β_5 is moved to $\Delta_6 = \{\beta_5\}$. The former would say that this set or any of its supper sets will never form a block. This is important to avoid unnecessary moves. The same thing happens at stage (12) where OPP retraces to stage (7) because he could not retrace to the stage (9). The current block β_{12} is set to $\{i, j\}$ which extends β_7.

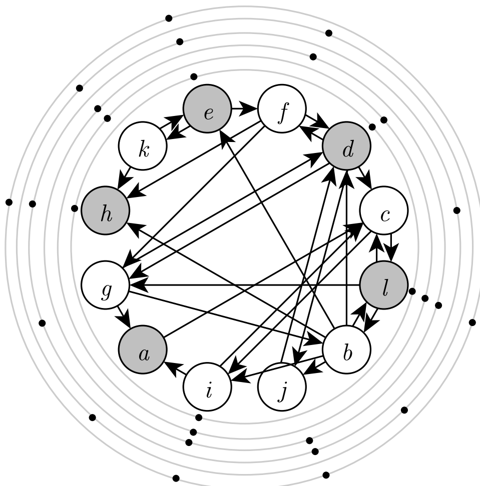
The dialogue continues until stage (15) where PRO plays a support move with argument l against which OPP could neither attack nor retrace to previous stages. At this stage the dialogue ends and PRO is declared as the winner.

The associated tree in Figure 4.2.(b) shows clearly the relation between the advanced arguments played by both parties. The tree in Figure 4.3.(b) is another dialogue tree for another dialogue where PRO is the winner. This can be easily observed since all leaf nodes are within an odd layer.

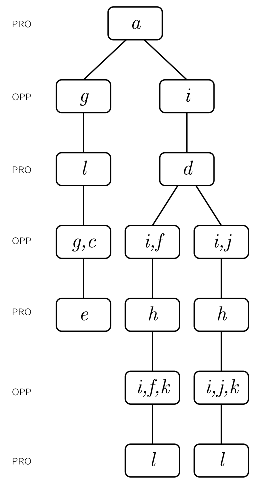
Let us now take an example where of another query $\mathcal{Q}'$ which is not universally accepted. The supporters are $\mathtt{S}(\mathcal{Q}') = \{a, d, e, h\}$. The dialogue is presented in Table 4.2 and its associated dialogue tree is shown in Figure 4.3.(a). In this example, OPP has been able to construct the block $\beta_6 = \{k, i, j\}$ in the last move which attacks all the supporters. This made PRO unable to continue the dialogue. Note that we do not allow retracing for PRO because one block is sufficient to prove the unacceptability of the query.

i	Move	π_i	h_i	θ_i	β_i
0	-	$\{a, d, e, h\}$	$\emptyset$	$\emptyset$	$\emptyset$
1	S(h)	$\{a, d, e\}$	$\{h\}$	$\{k, f, b\}$	$\emptyset$
2	C$(\{k\})$	$\{a, d\}$	$\{h\}$	$\{k, f, b\}$	$\{k\}$
3	S(a)	$\{d\}$	$\{a, h\}$	$\{k, f, b, g, i\}$	$\{k\}$
4	C$(\{k, i\})$	$\{d\}$	$\{a, h\}$	$\{k, f, b, g, i\}$	$\{k, i\}$
5	S(d)	$\emptyset$	$\{d, a, h\}$	$\{k, f, b, g, i, j\}$	$\{k, i\}$
6	C$(\{k, i, j\})$	$\emptyset$	$\{d, a, h\}$	$\{k, f, b, g, i, j\}$	$\{k, i, j\}$

Table 4.2: A dialectical proof for the non-universal acceptance of $\mathcal{Q}'$. Note that we omit Δ_i as it is always empty in this example.

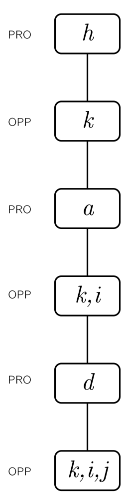


(a) The argumentation framework.

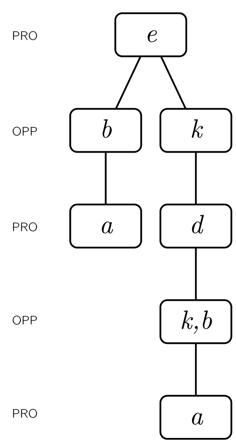


(b) The dialogue tree.

Figure 4.2: The circles are the extensions presented in an increasing order with $\mathcal{E}_1$ being the inner circle.



(a) Non-universal acceptance.



(b) Universal acceptance

Figure 4.3: The associated dialogue tree.

The question that one should ask is how we can be sure that if a query is not universally accepted then `OPP` will always win. The same thing when the query is universally accepted and `PRO` is the winner. In the next section we prove the completeness and soundness of the proof theory and present other interesting properties.

4.4 Properties

In this section we look at which proprieties our dialectical proof theory satisfies. Namely, finiteness, soundness and completeness. Next we shift to studying the dispute complexity defied in Dunne and Bench-Capon (2003) of dialectical proofs.

4.4.1 Finiteness, soundness and completeness

As indicated in Amgoud et al. (2013); Johnson et al. (2003), finiteness or termination is an important property for any dialogue, since a possibly infinite dialogue will fail to meet the intended goal, i.e. provide a proof in a finite way. In what follows we show how our dialectical theory produces always finite dialogues.

To establish such property we need to show that for any dialogue d its associated dialogue tree is finite. Such result can be established by showing that the height of the tree is finite and that for each node the number of its children is finite.

Lemma 4.4.1. Let $\mathcal{H}$ be an argumentation framework[4] and $\mathcal{D}^{\infty}$ be the set of all possible dialogues over $\mathcal{H}$. Given $\mathcal{T}(d) = (\mathcal{V}, \mathcal{D})$ of any $d \in \mathcal{D}^{\infty}$ the following hold:

1. $\exists k \in \mathbb{N}$ such that $\texttt{Height}(\mathcal{T}(d)) \leqslant k$.

2. $\forall\ v \in \mathcal{V}, \exists l \in \mathbb{N}$ such that $|\mathcal{C}(v)| \leqslant l$.

Such that $\texttt{Height}(\mathcal{T}(d))$ is the height of the tree $\mathcal{T}(d)$ and $\mathcal{C}(v)$ is the set of all child nodes of v.

Proof . *Let us suppose that* $\texttt{Height}(\mathcal{T}(d))$ *is infinite, and let P be the longest path in $\mathcal{T}(d)$ starting from the root node. This means either there are infinitely many supporting arguments used in P, or there are some infinity*

[4] Throughout the chapter, this refers to argumentation frameworks as defined in Chapter 3.

repeated supporting arguments used in P. The first one is impossible since we are dealing with finite argumentation frameworks (the set of all arguments is finite). The second is impossible since once an argument is played it cannot be advanced afterwards in the same path (see SE_1 of SUPPORT *move).*

Let us suppose that $|\mathcal{C}(v)|$ is infinite. This means that either (i) v is a supporting argument and it has infinitely many attacker; or (ii) v contains arguments that are advanced to attack previous supporters. The first case is impossible since the argumentation framework is finite, and the second is impossible since if it were the case then `PRO` *would be allowed to retrace against counter moves, which is forbidden in our framework.* ■

In Amgoud et al. (2013) an additional constraint has been added to finiteness, i.e. the finiteness of the moves' contents. This constraints insures that the arguments advanced within the dialogue are finite. In our context we distinguish two cases, (i) the argument in a support moves should be finite, and (ii) the set of arguments advanced in a counter move should be finite too. Fortunately, the two cases are verified in our argumentation framework because as shown in Chapter 2 the set of arguments $\mathcal{A}$ for any argumentation framework over a possibly inconsistent knowledge base is finite and the set of attacker for a given argument is finite. We get the following result on finiteness.

Theorem 4.4.1 (Finiteness)**.** Let $\mathcal{H}$ be an argumentation framework and $\mathcal{D}^\infty$ be the set of all possible dialogues over $\mathcal{H}$. Then, for every $d \in \mathcal{D}^\infty$, $\exists k \in \mathbb{N}$ such that $|d| \leqslant k$.

Proof . *Let us suppose that d is infinite. This means, either (i)* $\texttt{Height}(\mathcal{T}(d))$ *is infinite; or (ii) there is a node in $\mathcal{T}(d_n)$ with infinitely many child nodes. From Lemma 4.4.1, the two cases are impossible.* ■

Finiteness is not sufficient alone as a desirable property for a dialectical proof theory. After all, if a dialectical proof theory gives finite dialogues but incorrect results then such proof theory would be useless. Soundness is the property that insures that the proof theory gives only correct results. In other words, if one has a dialectical proof for universal acceptance (resp. non-universal acceptance) of a query then the query should be universally accepted (resp. not universally accepted).

Before proceeding to soundness let us show that the dialectical proof theory is *consistent* in the sense that there is no two dialogues about a query $\mathcal{Q}$ such that `PRO` wins one and loses the other. Put differently, if one of the participant wins a dialogue about a given query $\mathcal{Q}$ then we are sure that he will win all the other dialogues about $\mathcal{Q}$.

Proposition 4.4.1. Let $\Omega(\mathcal{H}, \mathcal{Q})$ be the set of all terminated dialogues about $\mathcal{Q}$ in $\mathcal{H}$ and let $d \in \Omega(\mathcal{H}, \mathcal{Q})$. Then, if d is won by PRO (resp. OPP) then so is all $d' \in \Omega(\mathcal{H}, \mathcal{Q})$.

Proof . *Suppose that d is won by* OPP *and there exists another dialogue d' that is won by* PRO*. This means that* OPP *has failed to construct a block in d''. This means that either (i) there is no block, or (ii) the Procedure 4.2.1 is not exhaustive. The former is in contradiction with the fact that* OPP *has won d therefore a block does exist. The latter is in contradiction with the evident fact that the procedure indeed tries all possible moves.* ■

This property is very important since we do not want to have a dialectical proof theory that is contradictory. It turns out that this property is important for soundness. In what follows soundness is characterized by the existence of one winning dialogue (by PRO or OPP).

Theorem 4.4.2 (Soundness)**.** Given a dialogue d_n about the query $\mathcal{Q}$, if d_n is won by PRO then $\mathcal{Q}$ is universally accepted.

Proof . *Let us proceed by contradiction. Suppose that d_n is won by* PRO *but $\mathcal{Q}$ is not universally accepted. On the one hand, recall that if $\mathcal{Q}$ is not universally accepted then there exists a block B against all $\mathcal{Q}$'s supporters. On the other hand, if* PRO *has won d_n then* OPP *could not find any block that attacks all supporters advanced in d_n. This means that either (i)* OPP *search was not exhaustive or (ii) there is no such block. As one can see, (ii) is in contradiction with the assumption and (i) is in contradiction with the fact that the Procedure 4.2.1 is exhaustive.* ■

If the dialectical proof theory is sound but does not provide dialectical proofs for all universally (resp. not universally) accepted queries then it would be incomplete. In what follows we provide a proof for completeness.

Theorem 4.4.3 (Completeness)**.** Given a query $\mathcal{Q}$. If $\mathcal{Q}$ is universally accepted then PRO wins any dialogue about $\mathcal{Q}$.

Proof . *By contradiction, if $\mathcal{Q}$ is universally accepted and* PRO *loses then* OPP *has constructed a block β_n for $\mathcal{Q}$. This means that $\mathcal{Q}$ is not universally accepted, which is a contradiction.* ■

4.4.2 Dispute complexity

In this subsection we are interested in the question of how many moves the dialogue would contain for a query (at best-case) to establish its universal acceptance (non-universal acceptance). Dunne and Bench-Capon (2003) have introduced the so-called dispute complexity for a given argument in a given argumentation framework. We adapt this definition to our context and we define the dispute complexity for a given query over a given instantiated argumentation framework as follows.

Definition 4.4.1 (Dispute complexity)**.** *Let $\mathcal{H}$ be an argumentation framework and $\mathcal{Q}$ be a query. Its dispute complexity $\delta(\mathcal{H}, \mathcal{Q})$ is defined as follows:*

$$\delta(\mathcal{H}, \mathcal{Q}) = min(|d| : d \textit{ is a terminated dialogue about } \mathcal{Q} \textit{ in } \mathcal{H})$$

It is read as the dispute complexity of the query $\mathcal{Q}$ in the argumentation framework $\mathcal{H}$.

The dispute complexity is the minimal number of moves that can be used to prove that $\mathcal{Q}$ is universally accepted or not universally accepted. Dunne and Bench-Capon (2003) have given an exact characterization of such complexity for credulous acceptance by considering as an input the argumentation framework and all admissible sets. Our goal in what follows is to propose some bounds for such complexity in universal (or non universal) acceptance. To proceed we need to recall the minimum hitting set problem.

Definition 4.4.2 (Minimum hitting set)**.** *Given a collection $\mathcal{C} = \{S_1, ..., S_m\}$ of finite nonempty subsets of a set $\mathcal{B}$ (the background set). A* hitting set *of $\mathcal{C}$ is a set $H \subseteq \mathcal{B}$ such that $S_j \cap H \neq \emptyset$ for all $S_j \in \mathcal{C}$. A minimum hitting set of $\mathcal{C}$ is the smallest hitting set for $\mathcal{C}$ with respect to set cardinality.*

In Definition 3.3.13, page 61 we have seen that a block of a given query is necessarily a hitting set.

Notation 4.4.1. Let $\mathcal{Q}$ be a query, $\mathcal{H}$ an argumentation framework such that $\mathcal{Q}$ is not universally accepted in $\mathcal{H}$ and $\mathcal{C} = \{\mathtt{range}^-(x) \mid x \in \mathtt{S}(\mathcal{Q})\}$:

- $HS(\mathcal{H}, \mathcal{Q})$ denotes the set of all hitting sets of $\mathcal{C}$.
- $MHS(\mathcal{H}, \mathcal{Q})$ denotes the set of all minimal hitting sets of $\mathcal{C}$.
- $BS(\mathcal{H}, \mathcal{Q})$ denotes the set of all blocks of $\mathcal{Q}$.
- $MinBS(\mathcal{H}, \mathcal{Q})$ denotes the set of all minimum blocks of $\mathcal{Q}$.

- The block number of $\mathcal{Q}$ in $\mathcal{H}$ is the size of the minimum block: $\tau(\mathcal{H},\mathcal{Q}) = min(|B| : B \in MinBS(\mathcal{H},\mathcal{Q}))$.
- The hitting set number is the size of the minimum hitting set of $\mathcal{C}$: $\alpha(\mathcal{H},\mathcal{Q}) = min(|S| : S \in MHS(\mathcal{H},\mathcal{Q}))$.

The block number corresponds to the minimum block which is the smallest block *(w.r.t set-cardinality)* among all blocks. Note that it is not necessary that every minimum hitting set of $\mathcal{C}$ is a minimum block, because a minimum block imposes that its members have to belong to the same admissible set (see Example 4.4.1 below). Therefore it is possible to have a block which is minimum but does not correspond to any minimum hitting set. In contrast, a minimum block has to be a hitting set. We get the following straightforward relations:

Fact 4.4.1. The following statement hold:

- $BS(\mathcal{H},\mathcal{Q}) \subseteq HS(\mathcal{H},\mathcal{Q})$.

From this fact we can easily deduce that the block number can be equal or greater than the hitting set number of a query in an argumentation framework.

Corollary 4.4.1. $\tau(\mathcal{H},\mathcal{Q}) \geqslant \alpha(\mathcal{H},\mathcal{Q})$.

In the context of a dialogue about a query $\mathcal{Q}$, the minimum block represents what the opponent would play in order to finish the dialogue as fast as possible. It means that the opponent will not use counter moves that might be avoided. Therefore, the dispute complexity of non-universal acceptance can be bounded by such number.

Theorem 4.4.4. For any terminated dialogue d about $\mathcal{Q}$ in an argumentation framework $\mathcal{H}$ where $\mathcal{Q}$ is not universally accepted:

$$\delta(\mathcal{H},\mathcal{Q}) = 2 \times \tau(\mathcal{H},\mathcal{Q}).$$

This theorem shows that given a query $\mathcal{Q}$ the complexity of the dispute in an argumentation framework is equal to the double of the block number. This is explained as follows, if the size of the minimum block B equals n then at each stage `OPP` will extend his current block by advancing one attacker at each stage. Therefore, for each SUPPORT move we will have a COUNTER move that extends the current block by one argument. When the current block reaches the size n, that means `OPP` has played all the arguments of the minimum block, PRO will have no supporting argument to advance, thus the dialogue terminates after $2 \times n$ stage. Note that if we allow the extension of the current block with more than one argument, which is not the case, then it is possible for OPP to play the whole minimum block in the first counter move, this would make the dialogue terminates after 2 moves.

Example 4.4.1. *Consider the example in Table 4.2, page 88.*

- $\mathtt{S}(\mathcal{Q}) = \{a, d, e, h\}$.
- *the set of all sets of attackers $\mathcal{C}$ is as follows:*

  1. $\mathtt{range}^-(a) = \{i, j\}$.
  2. $\mathtt{range}^-(d) = \{g, j, b, f\}$.
  3. $\mathtt{range}^-(e) = \{b, k\}$.
  4. $\mathtt{range}^-(h) = \{b, k, f\}$.

  The following minimum hitting sets are candidate blocks:

  1. $B_1 = \{b, i\}$.
  2. $B_2 = \{j, b\}$.

  As one can see these minimum hitting sets do not belong to any admissible set since they are not conflict-free. Hence the minimum blocks will have sizes of at least 3. Therefore, the following are minimum blocks (among others):

  1. $B_3 = \{k, i, j\}$.
  2. $B_4 = \{j, f, k\}$.
  3. $B_5 = \{i, f, k\}$.

  In the dialogue d_6, B_3 has been constructed. We can see clearly that: $\delta(\mathcal{H}, \mathcal{Q}) = 2 \times 3 = 6$.

A direct result of the theorem is the following.

Corollary 4.4.2. Let d be the shortest dialogue for the non-universal acceptance of $\mathcal{Q}$. The associated dialogue tree $\mathcal{T}(d)$ is a chain.

This result is straightforward since OPP will attack all supporters without any need to retrace, hence there will be no branching in the associated dialogue tree (see Figure 4.3.(a)).

Let us turn to the dispute complexity for universal acceptance. As we have seen in Chapter 2, proponent sets also have a strong relation with minimal hitting sets, in fact they are minimal hitting set over the reduct of extensions (see Definition 3.3.12, Chapter 3). We define similarly the proponent number and the attack degree of a query in an argumentation framework to be able to bound the dispute complexity of its universal acceptance.

Notation 4.4.2. Let $\mathcal{Q}$ be a query and $\mathcal{H}$ an argumentation framework such that $\mathcal{Q}$ is universally accepted in $\mathcal{H}$.

- the proponent number is the size of the minimum proponent set: $\rho(\mathcal{H}, \mathcal{Q}) = min(|S| : \textit{ S is a proponent set of } \mathcal{Q} \textit{ in } \mathcal{H})$.
- the attack degree of $\mathcal{Q}$ in $\mathcal{H}$: $\deg(\mathcal{H}, \mathcal{Q}) = max(|\mathtt{range}^{-}(a)| : a \in \mathcal{P}(\mathcal{Q}))$ such that $\mathcal{P}(\mathcal{Q})$ is the set all supporting arguments that belongs to at least one minimum proponent set.

The proponent set is a minimal set of supporting arguments which covers the whole extensions. A minimum proponent set is the smallest proponent set with respect to set cardinality. When PRO is engaged in a dialogue he always uses the set of all supporters $\mathtt{S}(\mathcal{Q})$. It is obvious that a proponent set (minimum or not) can replace $\mathtt{S}(\mathcal{Q})$ since it represents all what PRO needs to establish the universal acceptance of $\mathcal{Q}$.

Proposition 4.4.2. Let P be a proponent set of a given query $\mathcal{Q}$ in an argumentation framework $\mathcal{H}$. Let us change the rules of the dialogue described earlier so that PRO plays only from P instead of $\mathtt{S}(\mathcal{Q})$. Then, the soundness and completeness are preserved.

The proposition is straightforward, because if there is no P the dialogue will not start and the query will not be accepted. If P exists then there is no block that can attack all members of P. Certainly, it is not intuitive to start with a proponent set because the goal of the dialogue is to establish whether such set exists or not. However, this result is very important to determine the smallest dialogue to prove $\mathcal{Q}$, consequently to bound the dispute complexity.

It is obvious that dialogues where PRO plays with minimum proponent sets are shorter than all other dialogues. Because in the former dialogues PRO will play only the support moves that are needed to terminate the dialogue.

Proposition 4.4.3. Let $\delta(\mathcal{H}, \mathcal{Q})$ be the dispute complexity of the query $\mathcal{Q}$ in $\mathcal{H}$. Let $\Theta(\mathcal{H}, \mathcal{Q})$ be the size of the shortest dialogue where PRO plays only with a minimum proponent set. Then,

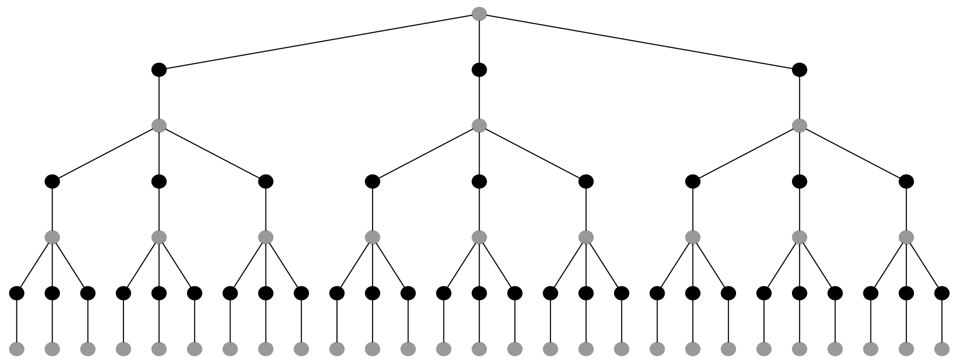


Figure 4.4: A worst-case associated dialogue tree with $k = 3$. Gray-colored nodes are support move nodes while black-colored nodes are counter move nodes.

$$\delta(\mathcal{H}, \mathcal{Q}) \leqslant \Theta(\mathcal{H}, \mathcal{Q}).$$

But this claim is not precise enough to serve as an upper bound. Actually, we need to bound the quantity $\Theta(\mathcal{H}, \mathcal{Q})$. To do so, one has to see what would be the worst-case size of such quantity.

If we imagine the associated dialogue tree of the shortest dialogue, the worst-case for such quantity would be that at each support node we would have k child with k being the worst-case number of attackers, which corresponds to the attack degree of the query $\mathcal{Q}$ in $\mathcal{H}$. This tree is similar to the one in Figure 4.4 (called the worst-case associated dialogue tree). We have the proponent number is 4 (number of gray nodes) and $k = 3$, therefore every support node has exactly 3 children, whereas every counter move has exactly one child. It is clear that to compute the worst-case quantity that bounds $\Theta(\mathcal{H}, \mathcal{Q})$ one needs to compute the number of nodes of the worst-case optimal associated dialogue tree. Therefore, we get the following theorem:

Theorem 4.4.5. For any dialogue d about $\mathcal{Q}$ in an argumentation framework $\mathcal{H}$ where $\mathcal{Q}$ is universally accepted:

$$\delta(\mathcal{H}, \mathcal{Q}) \leqslant 2 \times \frac{k^{h/2+1} - 1}{k - 1} - 1$$

Such that $k = \deg(\mathcal{H}, \mathcal{Q})$ and $h = 2 \times (\rho(\mathcal{H}, \mathcal{Q}) - 1)$.

The quantity h is the height of the worst-case optimal associated dialogue tree and $k = \deg(\mathcal{H}, \mathcal{Q})$ is the worst-case branching factor. Consider the example of Figure 4.4, the dispute complexity is bounded by the number of nodes in this tree:

$$2 \times \frac{3^{6/2+1} - 1}{3 - 1} - 1 = 1 + 3 + 3 + 9 + 9 + 27 + 27 = 79.$$

Let us see how we compute the upper bound of the dispute complexity on a real example.

Example 4.4.2. *Consider the example of Figure 4.3.(a) where we consider another query with* $\mathtt{S}(\mathcal{Q}'') = \{a, d, e, l, h\} \cup \{k\}$ *such that* k *is also a supporter of* $\mathcal{Q}$*. Note that we add* k *here to make the computation easier.*

- *the query* $\mathcal{Q}''$ *has two minimum proponent sets:* $\mathcal{P} = \{P_1, P_2\}$ *and* $P_1 = \{k, h\}$*,* $P_2 = \{k, e\}$*.*

Let us compute the upper bound for the dispute complexity of $\mathcal{Q}''$*.*

$$\delta(\mathcal{H}, \mathcal{Q}'') \leqslant 2 \times \frac{k^{h/2+1} - 1}{k - 1} - 1$$

Such that:

- *the proponent number* $\rho(\mathcal{H}, \mathcal{Q}'') = 2$*.*
- *the height of the worst-case associated dialogue tree* $h = 2 \times (\rho(\mathcal{H}, \mathcal{Q}'') - 1) = 2 \times (2 - 1) = 2$*.*
- *The query has two minimum proponent sets* $\mathcal{P} = \{\{k, h\}, \{k, e\}\}$*. The attack degree of* $\mathcal{Q}''$ *is* $max(1, 2, 3) = 3$ *for* k, e, h *respectively.*

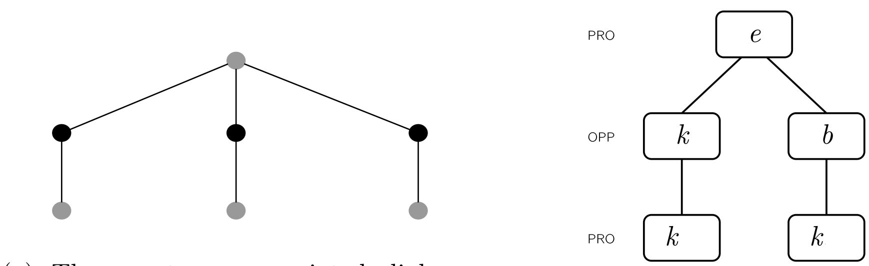
Therefore,

$$\begin{aligned} \delta(\mathcal{H}, \mathcal{Q}) &\leqslant 2 \times \frac{3^2 - 1}{3 - 1} - 1 \\ \delta(\mathcal{H}, \mathcal{Q}) &\leqslant 7 \end{aligned}$$

The worst-case associated dialogue tree is presented in Figure 4.5.(a). The shortest dialogue is presented in Figure 4.5.(b). The real dispute complexity is equal in this case to:

$$\delta(\mathcal{H}, \mathcal{Q}) = 5 < 7$$

In the next section we turn to the practical effects of meta-level dialectical explanations where we study their explanatory power and their effects on users.



(a) The worst-case associated dialogue tree for $\mathcal{Q}$.

(b) The associated dialogue tree for the shortest dialogue.

Figure 4.5: The two associated dialogue trees.

4.5 Effects of Meta-level Dialectical Explanations

The goal of meta-level dialectical explanations is to help the user to understand query entailment in inconsistent knowledge bases. In our case meta-level dialectical explanations are dialectical proofs of universal and non-universal acceptance. In this section we want to see whether these explanations would achieve such goal or not. As understanding is a vague concept and sometimes subjective. We focused on measuring the effect of Meta-level Dialectical Explanations on the accuracy (when answering certain questions) and the subjective evaluation of the clarity of explanations. Our claim is that if understanding has really taken place then the users would be more accurate in answering some questions and would appreciate better the meta-level dialectical explanations. As we will explain later, the experiment has been conducted on two groups, one has received meta-level dialectical explanations and the other has received one-shot argument-based explanations. One-shot argument-based explanations are those which explained Definitions 3.3.12 and 3.3.13 (page 61). Recall that a one-shot argument-based explanation for the non-universal acceptance of a query is a block, and a one-shot argument-based explanation for the universal acceptance of a query is a proponent set.

4.5.1 Method

In experimental settings within-subjects design is commonly used. In this design every participant is subjected to every single treatment one after another. Next, we compare the results for each treatment. This helps in eliminating participants individual differences. However, the problem with such design is the carryover effects where treatments can influence each other. To avoid such problem, between-subjects design is used where participants are divided into two (or more) groups, a control group and a treatment group(s).[5] Like that every participant is only subjected to a single treatment. In our experiment we choose the between-subjects design to avoid the carryover effects. One disadvantage of this design is the possibility of having individual differences. To minimize this effect we conducted the experiment with students from the same class. Let us see more details about the experiment.

Objective. The goal is to show the effect of meta-level dialectical explanations and one shot explanations on user's understanding of query entailment in inconsistent knowledge base under CQA. Our hypothesis is that meta-level dialectical explanations help better understanding the process of entailing a query from an inconsistent knowledge base and this would have a positive effect on **accuracy, time and explanations' evaluations.**

Subjects. The experiment is conducted with 34 subjects with an average age of 19. The subjects are second year computer science students at the university and they are not familiar with logic and argumentation. Therefore it is safe to assume that their abilities are similar. The experiment has taken place in IUT Montpellier during the last week of May 2016.

Materials and procedure. We developed a web application to perform the experiment.[6] Subjects are split into two equal groups (17 per group) with a random assignment. The control group (Group OE) and the treatment group (Group DE). Since the subjects are not familiar with logical formulas, knowledge bases are presented in a textual form by staying as faithful as possible to the underlying logical formalism (i.e. they contain facts, rules and constraints). Each fact, rule and negative constraint has been expressed in plain French text (e.g. a negative constraint: "a person cannot wear sunglasses and protective glasses in the same time").

All the subjects are presented with 7 different descriptions of certain situations containing inconsistencies (Figure 4.6 in 'Situation'). To avoid the effect of a priori knowledge, these situations where fictitious. For each situation all subjects are presented with a query (called *normal query*) and

[5] a.k.a independent measures design.

[6] The platform is available at `http://cloud.lirmm.fr/`. It is programmed by Abdelraouf Hecham.

its answer over the situation in question (we have in total 7 normal queries for 7 situations). The difference between the two groups is that one group (Group DE) has received only meta-level dialectical explanations and the other (Group OE) has received only one-shot argument-based explanations for the 7 queries (Figure 4.6 in 'Query and Explanation'). To variate the type of queries, 4 out of 7 are not universally accepted and the rest are universally accepted. Each subject is asked to do two things:

- Evaluate the explanations on the following scale: "not clear at all", "not clear", "clear", "so-so", "very clear" (Figure 4.6 in 'How do you find the explanation?").
- Answer by yes or no about the acceptance of a *test query* and give a justification for the answer. Note that we do so to test whether he/she really understood how (non-)universal query acceptance works.

In each group, the subject starts randomly with a situation as shown in Figure 4.6. The subject is given an unlimited time to understand the answer of the normal query (with the aid of the explanation) and to give his/her evaluation of the explanation. At this point, the test query is hidden, and when the subject clicks on 'Show test query' the test query is shown and the subject is asked to answer by yes or no in a time frame of 2 min. Next, he/she is asked to provide a justification in another 2 min. The justification is used to insure that the subjects answer seriously to the test query. Then he/she proceeds sequentially as described above until he/she terminates all the 7 situations. Once the subject terminates a situation he/she can never return and read or alter his/her answers or justifications. Note that the subjects cannot know whether their answers are right or wrong.

4.5.2 Results and analysis

Three variables have been considered, the accuracy of answers for test queries, the time taken to answer them and the scale of the evaluation of the explanations.We investigate the difference between the two groups with respect to these variables to show the effect of meta-level dialectical explanations on the subjects. In what follows we give the results for each variable with the discussion of the result.

4.5.2.1 Accuracy

In the accuracy we are interested in knowing the average accuracy of each subject in answering test queries. It is defined as the proportion of right

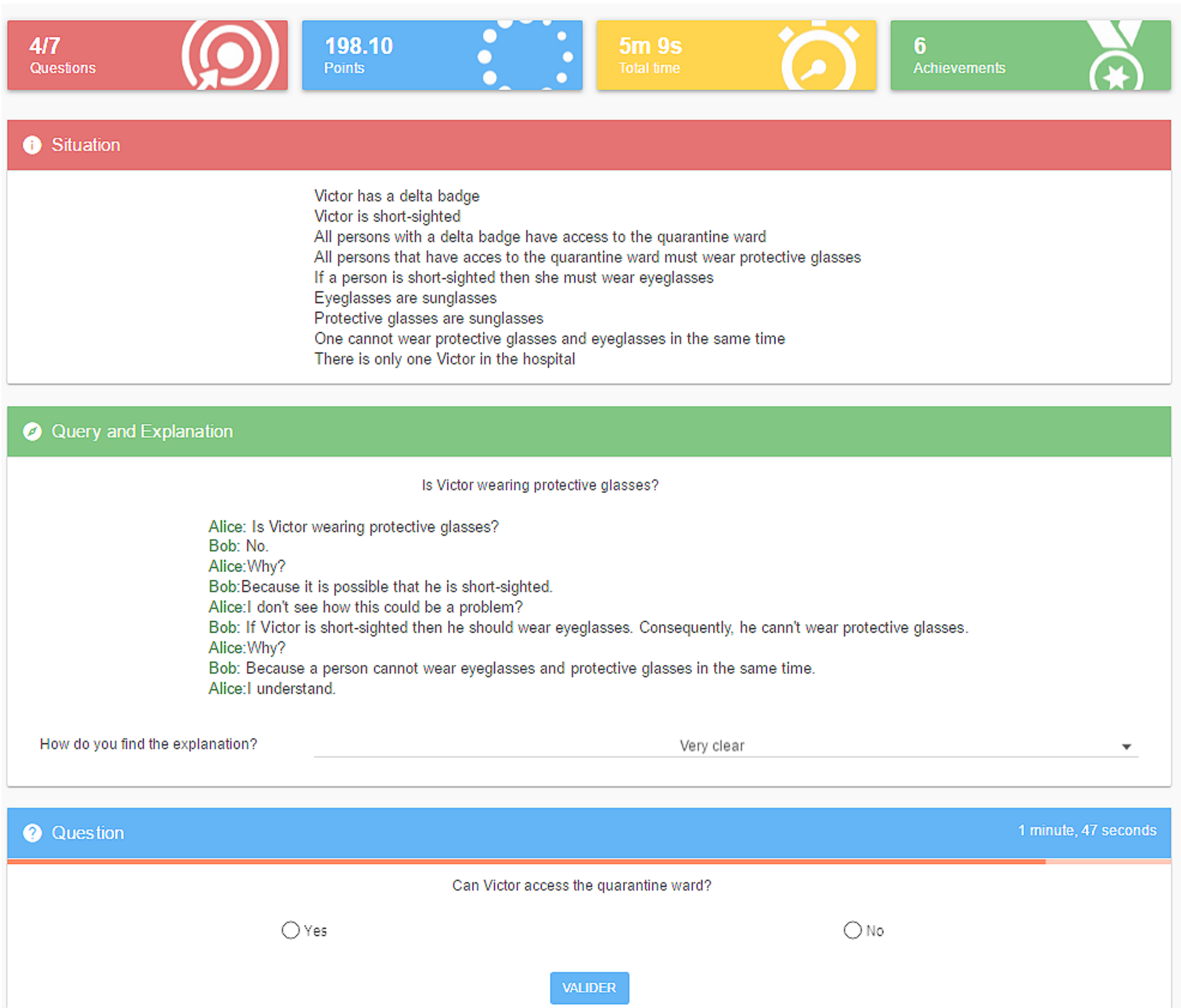


Figure 4.6: A screen shot of the experiment platform with a meta-level dialectical explanation (Group DE). Note that the platform is originally in French.

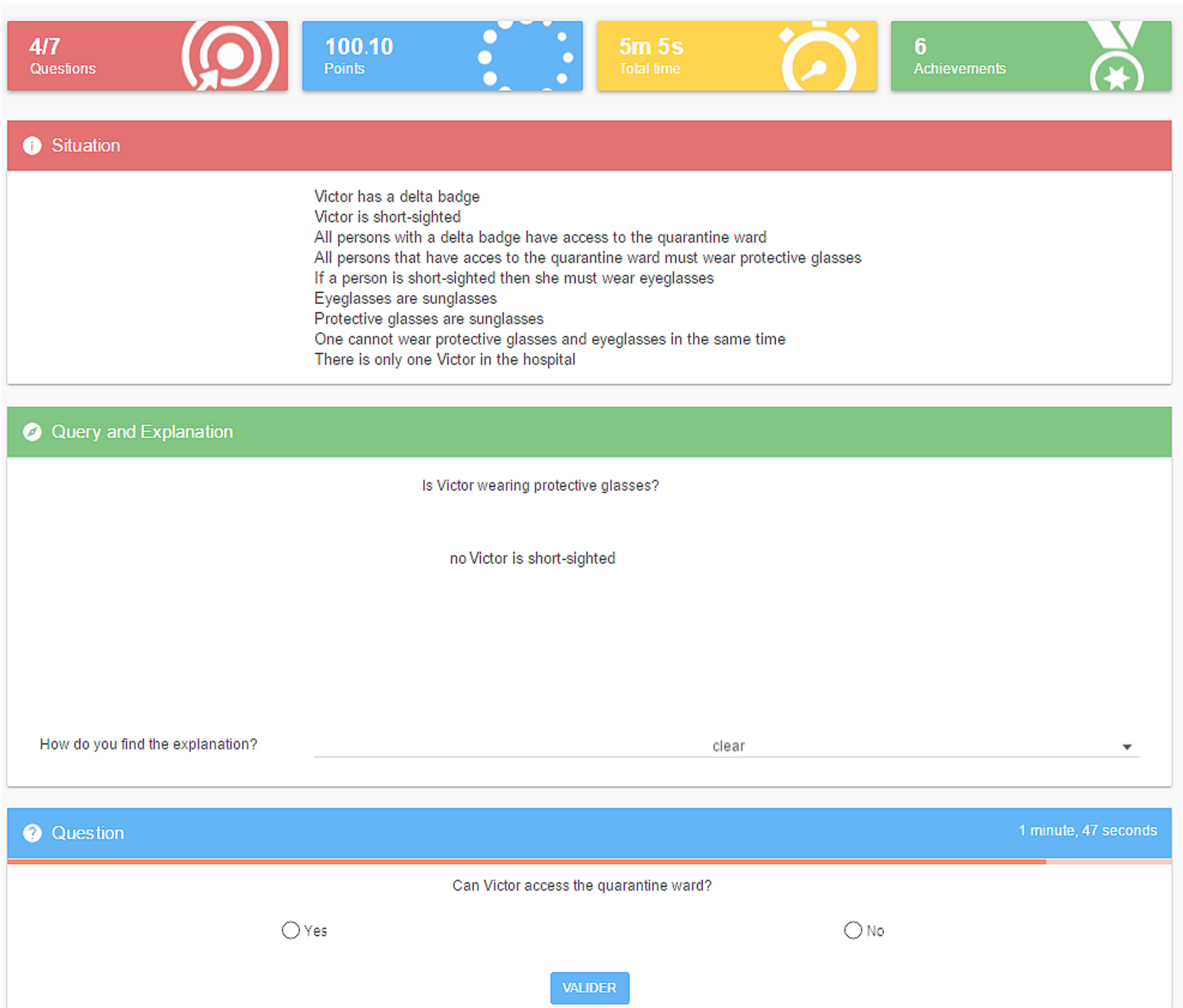


Figure 4.7: A screen shot of the experiment platform with a one-shot argument-based explanation (Group OE).

answers among the total answers of test queries (i.e. 7). After computing accuracy for each subject we compute the average accuracy of each group. Our hypothesis is that:

> *Group DE (Meta-level Dialectical Explanations) has a better accuracy than Group OE (One-shot Argument-based Explanations).*

Table 4.3 shows the average accuracy for each subject in each group. An overall look shows a close accuracy within groups and a significant difference between groups. This is confirmed by an independent-sample Mann-Whitney U test to check the positive effect of dialectical explanations on the subject's accuracy (p-value = 0.002), we can see that dialectical explanations are associated with an improved accuracy (group DE mean=0.77, group OE mean=0.63).[7] The mean is read as: on average, subjects in Group DE get 77% of the answers for test queries correct and on average subjects in Group OE get 63% of the answers correct. By computing the median on the number of correct answers per subject we find that for Group DE on average, subjects get 6 out of 7 correct answers while subject of Group OE get 4 out of 7 correct answers. The results in the table are ordered, we can observe that the accuracy is better in absolute term in Group DE. One can observe for instance that the third in Group DE is at least as good the first one in Group OE.

To confirm the significance we ran a chi-square ($\chi 2 = 5.801$, p-value=0.016 < 0.05) which indeed has shown a significant difference between Group DE and Group OE. Table 4.4 shows the contingency table of correct and incorrect answers for each group. We can observe that the number of correct answers for Group DE is bigger than the one of Group OE. The same thing can be observed for the number of incorrect answers which is almost the half of those of Group OE. In conclusion, the data **validates the hypothesis** as it shows in general that explanations (dialectical or one-shot) are helpful in improving accuracy. However, it tells more specifically that dialectical explanations contribute better to this improvement than one-shot argument-based explanations.

[7] Mann-Whitney U test is a statistical hypothesis test which tells whether the means of two groups are significantly different from each other. The p-value refers to the probability that the difference we are observing is not significant (i.e. due to chance). The lower it is the more significant the difference is. It should be less than 0.05. This test is used because our data is not normally distributed the student t-test is used when the distribution is normal.

	Group DE	Group OE
1	1	0,85
2	1	0,85
3	0,85	0.71
4	0,85	0.71
5	0,85	0.71
6	0,85	0.71
7	0,85	0.71
8	0,85	0.71
9	0,85	0.57
10	0.71	0.57
11	0.71	0.57
12	0.71	0.57
13	0.71	0.57
14	0.71	0.57
15	0.71	0.42
16	0.42	0.42
17	0.42	0.42
Avg	0,77	0.63
Std	0.16	0.13

Table 4.3: Average accuracy per subject for each group. Please note that the experimental design is between-subjects that means the accuracies in the same row are for different subjects.

4.5.2.2 Answer time

The time variable here is considered with respect to the rapidity of answering test queries. Our hypothesis is that:

> *In the presence of Meta-level Dialectical Explanations, subjects of Group DE would understand better than those of Group OE (One-shot Argument-based Explanations). As a consequence, Group DE answer time would be shorter than the one of Group OE.*

An independent-sample t-test shows no significant difference (p(33) = -1.4739, p-value = 0.07744 > 0.05) with Group DE mean = 10.9 sec and Group OE mean = 16.1 sec, a difference of approximately 6 sec which **does not validate the hypothesis**. An interpretation of this non-significance is

	Correct	Incorrect	Total
Group DE	92	27	119
Group OE	75	44	119
Total	167	71	238

Table 4.4: Contingency table of accuracy for Group DE and Group OE.

that many factors are included which make the difference seems to be a mere chance rather than a real impact of the type of explanation. For instance, the subject can be prudent therefore he/she would take more time in verifying the response. This could remarkably result in a delay of answering test queries.

4.5.2.3 Evaluation of explanations

In this variable we look at any difference between the evaluation of explanations by subject. Our hypothesis is that:

Meta-level Dialectical Explanations are more intelligible than One-shot Argument-based explanation.

Recall that each subject in each group has been asked to evaluate the explanation he/she received on a scale of 1 to 5 (i.e. "not clear at all", "not clear", "so-so","clear", "very clear") with 1 being "not clear at all" and 5 as "very clear". Each subject has evaluated 7 explanations, giving 119 data entries for each group.[8] Since the data is ordinal (i.e. ranks) we performed an independent-sample Mann-Whitney U test (U=5553.5, p-value = 0.0041 < 0.05) which shows a significant difference between Group DE and Group OE (median Group DE = clear, median Group OE = so-so). We ran a chi-square ($\chi^2 = 23.1768$, p-value=0.0001 < 0.05) which has also shown a significant difference between the two groups. Figure 4.8 shows the bar chart of the evaluations for each group. The graph shows that for Group DE the number of *not clear at all* and *not clear* is significantly less than those of Group OE. Moreover, although being close on *so so* and *clear*, Group DE differs significantly on *very clear*. In conclusion, the data point to the fact that Meta-level Dialectical Explanation are more intelligible than One-shot Argument-based Explanations which **validates the hypothesis**.

[8]Number of subjects in each group multiplied by the number of explanations, i.e. 17 $\times$ 7=119.

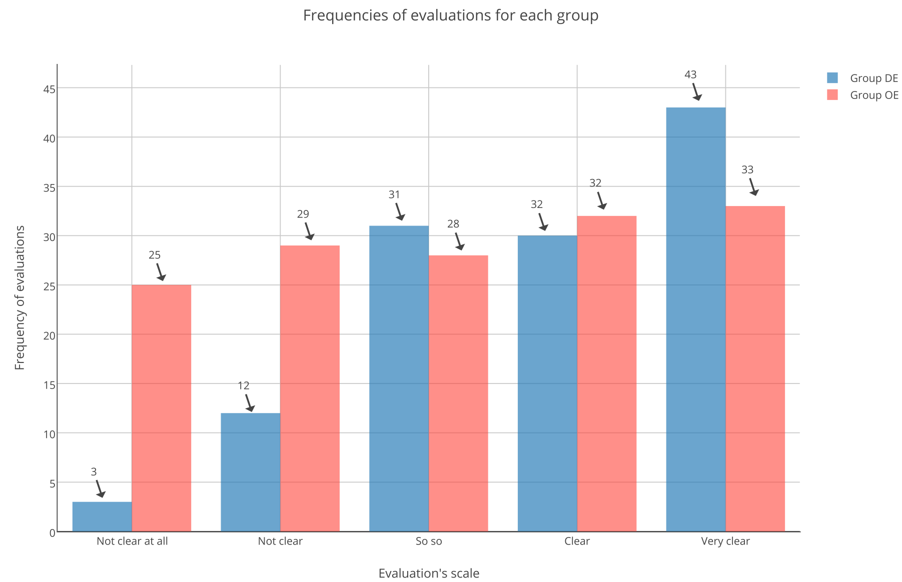


Figure 4.8: The bar-chart of the evaluation of explanations for Group DE and Group OE.

4.5.3 Post-hoc analysis

In this subsection we are interested in looking at the data and extract some recurring patterns. Although these patterns were not predicted to occur a priori to the experiment, we find it important to report them and give some interpretations.

Working on the same data entries we looked at the *justification length* variable for each group. First we looked at the distributions of justification length for each test query in each group. So, we have a total of 7 test queries for which the answer should be justified. Therefore, we get 7 justifications per person. This makes a total of 119 data entries for each group. We looked first at the average length of justifications in Group DE and Group OE. An independent-sample Mann-Whitney U test ($U = 4284.5$, p-value$= 0.00012 < 0.05$) shows a significant difference between the groups with respect to the lengths. The mean for Group DE is 89.31 (standard deviation 57.39) and for Group OE is 64.92 (standard deviation 44.46). Figure 4.9 shows the box-plot of the two groups. As one can notice from the plot and the standard deviations, the distribution of lengths in Group DE is more dispersed than Group OE where more points above 150 are found in Group DE.

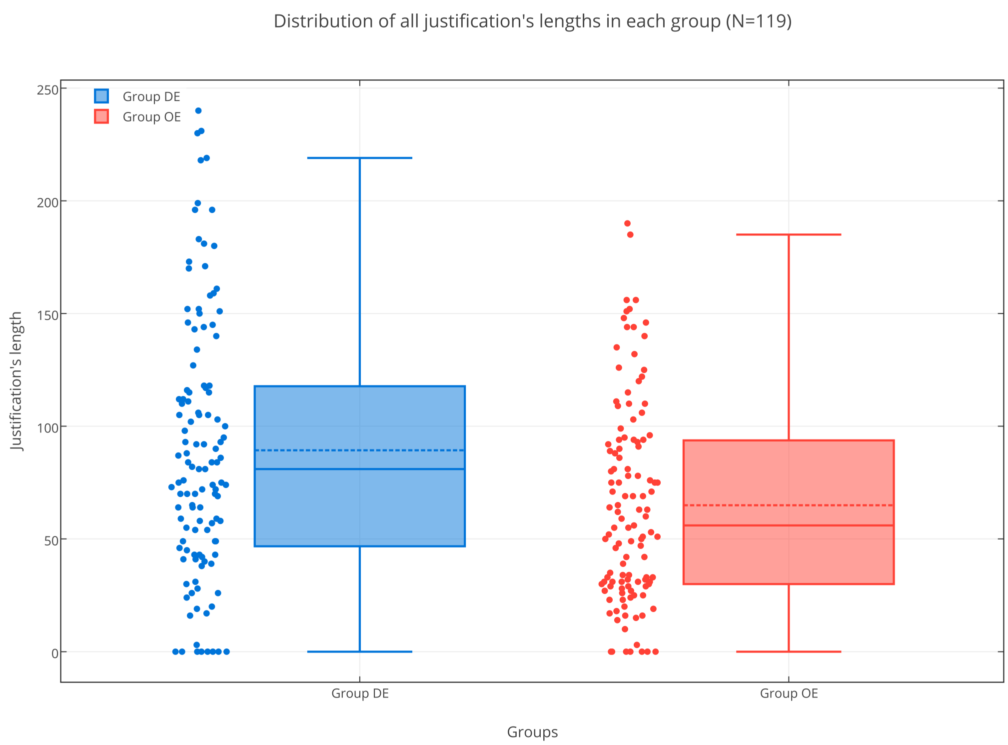


Figure 4.9: Box plot for each group where points represent justification's length. Precisely, each point is the length of a justification for a test query. Total data entries for each group is N=119. The dashed line is the mean 89.31 for Group DE and 64.92 for Group OE.

To have more insight on justifications' length we looked at the average length per test query for each group. That means we computed for each test query the average length of justifications provided by the subjects in each group.[9] Figure 4.10 gives the bar-chart for each test query. We ran an independent-sample Mann-Whitney U test to see whether the difference is significant. The result gives a p-value equals to 0.013 (N=7) which is significant. One notable observation is the difference between the averages of DE and OE for Q5 which is remarkable. After observing data, we concluded that the subjects of Group OE in Q5 followed the same pattern (length and structure) of the explanation they had received for the normal query 5. In fact, the normal query 5 has the shortest one-shot argument-based explanation (26 characters) among all 7 normal queries. Whereas, it has a relatively long meta-level dialectical explanation (515 characters). To give more insight on that point, Table 4.5 shows the normal query 5 and test query 5 alongside some justifications by different subjects from Group OE.

Normal query 5:	Is Jude wearing sunglasses?
Answer:	No
One-shot explanation:	Because Jude is a puma
Test query 5:	Is Jude wearing running shoes?
Answer:	No.
Subject's A justification:	Because Jude is a puma. (15 sec)
Subject's B justification :	No, because Jude is a puma. (11 sec)
Subject's C justification :	Jude is a serpent. (9 sec)

Table 4.5: Situation 5 with some samples of justifications. The seconds refer to the taken time to write the justification by the subjects of Group OE).

Although other subjects in Group OE have answered with longer justifications (for different test queries), in the overall all of them have answered with relatively short justifications compared to Group DE. In what follows we give two hypotheses to explain why the average length for Group OE is bigger than Group OE. We give some future directions to verify such hypotheses and some threat to validity.

[9] In fact, justifications are attached to the answers of the test queries, but for brevity we refer to test queries instead of their answers as it is unambiguous.

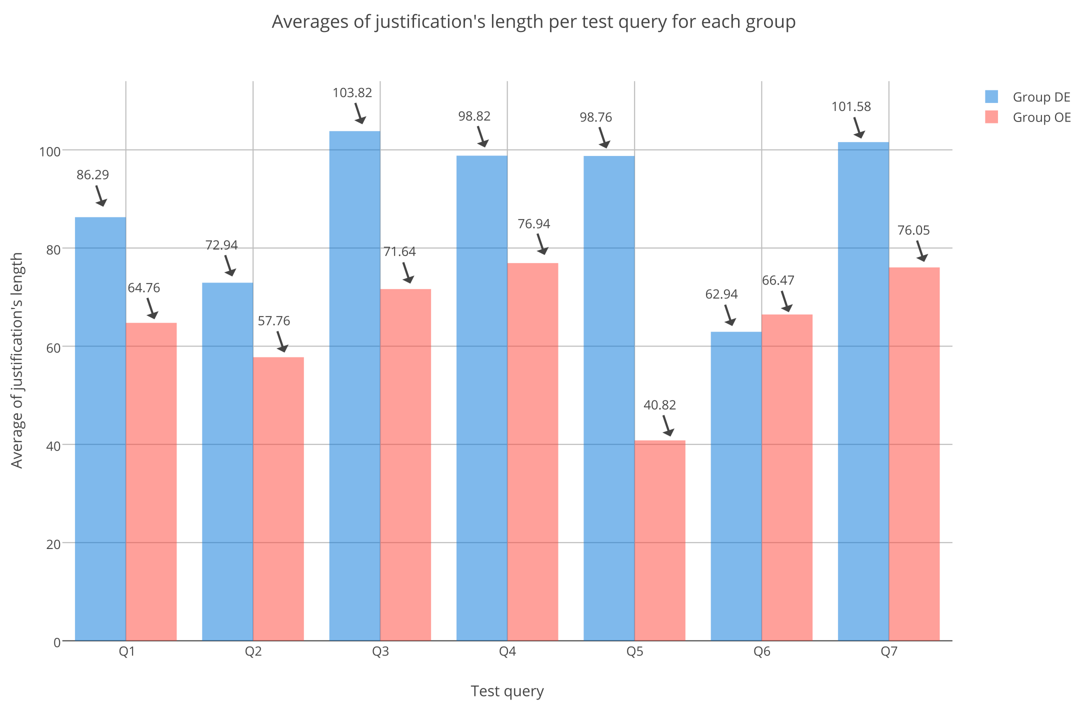


Figure 4.10: A bar-chart showing averages of justification length per test query for each group. The averages are computed over a total data entries of N=119 for each group.

Hypothesis 1
Subjects provide justifications with length proportional to the length of the explanations they receive.

In this hypothesis we claim that the subjects when they receive short (resp. long) explanations they reply with short (resp. long) justifications. This happens because they are just following what the experiment provides. Therefore, the length of justification is not related to the type of explanation but rather to its length. Our data cannot serve as an evidence to validate the hypothesis because of the risk of having individual differences. To validate such hypothesis one has to augment the number of subjects and test queries and vary the length of explanations. In addition, a within-subject design would be more appropriate where the same subjects undergo the same experiment under two conditions, i.e. with one-shot argument-based explanations and with meta-level dialectical explanations. By doing so, we would be sure that the length is not due to some differences between subjects since each subject will be observed under two conditions.

Another alternative hypothesis is that the type of explanation would influence the length of justification.

Hypothesis 2
Subjects who received Meta-level Dialectical Explanations understood well the explanations therefore they try to better justify by providing as much information as possible.

Although a quantifiable measure is not defined, we observed in the justifications of Group DE some kind of *elaboration* which is absent in Group OE. This elaboration aims at making the justification as clear as possible. For instance, take the test query 5 presented above in what follows we present some justifications from Group DE.

- Justification of subject E: Jude does not wear running shoes because it is possible that Jude is a serpent and given that Jude cannot be a puma and a serpent so Jude does not wear running sh. (60 sec)
- Justification of subject F: Jude can be a serpent. Except that a serpent does not wear running shoes. But Jude can be a puma. So as we are not sure, it is safe to say that she is not wearing running shoes. (50 sec)

This pattern is common among subjects of Group DE for all test queries. We can see that it is more elaborated and logical. The subjects in the two justifications above have taken all the time they need to provide a concise justification. In fact they reached the time limit which is 60 sec. One can see that subject E could not even finish the phrase due to time limit ("sh" should be "shoes"). However, this evidence is not enough to validate the hypothesis. Actually, we still need a within-subjects experiment to be able to draw more reliable conclusions.

To conclude the section, the experiment we have carried on shows that meta-level dialectical explanations (compared to one-shot explanations) have a positive impact on accuracy. Moreover, they are judged to be more clear than one-shot argument-based explanations. However, the experiment shows that dialectical explanations do not have a significant impact on the time of response to test queries.

4.6 Conclusion

In this chapter we have provided a dialectical proof theory for universal acceptance in the preferred/stable semantics. We have proved its completeness and soundness. We have also shown that such theory is consistent and does not provide contradictory results.

- Theorem 4.4.1 (Finiteness): the dialogues of the dialectical proof theory always terminate.
- Proposition 4.4.1: there is no two dialectical proofs for $\mathcal{Q}$ where one of them is won by `PRO` and the other is won by `OPP`.
- Theorem 4.4.2 (Soundness): when `PRO` wins (resp. loses) then $\mathcal{Q}$ is universally accepted (resp. not universally accepted).
- Theorem 4.4.3 (Completeness): when $\mathcal{Q}$ is universally accepted (resp. not universally accepted) then `PRO` wins (loses).

We have also studied the dispute complexity of dialectical proofs and we have provided the following results:

- Theorem 4.4.4: given a non-universally accepted query $\mathcal{Q}$ in an argumentation framework $\mathcal{H}$ then `OPP` will establish its non-universal acceptance in a dialogue of size equals to $2 \times \tau(\mathcal{H}, \mathcal{Q})$.

- Theorem 4.4.5: given a universally accepted argument $\mathcal{Q}$ in an argumentation framework $\mathcal{H}$ then PRO will establish its universal acceptance in a dialogue of size no bigger than 2 times the number of nodes in the worst-case associated dialogue tree of $\mathcal{Q}$.

A comment about these results is in order. It is clear that the bounds proposed for the dispute complexity of (non-)universal acceptance are estimated in terms of the proponent and the block numbers. Unfortunately, these numbers are not given as inputs and they should be computed. It is obvious that computing such numbers is hard to compute, in fact they are equivalent to finding a minimum proponent/block for the query, which would solve the problem in the first place. The good news is that these numbers are related to the hitting set problem, therefore if one can estimate the cardinality of the minimum hitting set we can easily estimate the proponent or the block number by just considering the number of arguments and attack degree as inputs in (non-)universal acceptance. This can be achieved by using the results from Eustis (2013) on the independence number in hypergraphs which is the complement of the hitting set number (also known as the transversal number).

On the practical side, we have evaluated the effect of meta-level dialectical explanations on users. We have found that while they have no significant effect on answer time, they have indeed a clear effect on the understanding of query acceptance which results in an improvement of accuracy. The experiment has also shown that meta-level dialectical explanations are clearer than one-shot argument-based explanations. The post-hoc analysis revealed an interesting pattern about the justification length and structure. It has shown that subjects who received meta-level dialectical explanations provided, on average, lengthy justifications with concise logical structure. Whereas those subjects who received one-shot argument-based explanations gave, on average, shorter justifications without a logical structure.

Dialectical proof theories have an interesting explanatory power, they exhibit the reasoning procedure in a dialogical form between a proponent and an opponent. In interactive systems, a dialectical proof theory can be used to explain entailment of queries in inconsistent knowledge bases where the user can play the role of the proponent or the opponent in order to understand entailment. However, this type of explanations is not sufficient when the user asks for domain-specific explanations. More precisely, explanations about the content of the domain represented within the knowledge base. In the next chapter we present a formal dialogue model of explanations equipped with argumentative and explanatory speech acts and a formal protocol.

5 Object-level Dialectical Explanations

In this chapter we are interested in *object-level dialectical explanations*. As opposed to meta-level dialectical explanations, object-level dialectical explanations are domain-specific explanations which are meant to use the domain knowledge stored in the knowledge base to answer the user's requests for explanations. These explanations are not "one-shot" but rather an exchange in form of a dialogue (hence *dialectical*) between the `User` and the `Reasoner` to arrive at the goal of understanding transference. In Section 5.1 we start by a simple overview on dialogue models developed in the literature. Then, in Section 5.2 we give a motivating example that presents the general skeleton of an explanation dialogue that we aim at formalizing. In Section 5.3 we present how the state of the art explanation dialogue model fails to capture certain desirable aspects. For that matter, in Section 5.4 we propose a dialogue model for explanations called `EDS`. We formally define its protocol's syntax and semantics and we investigate the role of commitments in such dialogue and their relation to termination and success. In Section 5.5 we discuss how this dialogue model can be extended with argumentative faculties so that it can account for users' objections against explanations. Next, in Section 5.6 we provide the formalization of the motivating example in the dialogue model `EDS`. Finally, in Section 5.7 we present a use case within the DUR-DUR project that shows how object-level dialectical explanations can be used in knowledge acquisition and inconsistency resolution.

5.1 Introduction

Communication takes a big part in human-human interactions. Dialogue lays in the heart of such communication. In dialogues, humans take turn in uttering natural language sentences with some preset goals. One may say a joke to soften the mood or advance an argument to convince the hearer. Different sentences, or let us say *utterances*, perform different speech acts

(Austin, 1975). These speech acts produce, under specific circumstances, changes in the observable world and the mental state of the hearer. For instance, the utterance "the drink is really cold!" is performing the speech act of *complaint* in a winter day. This speech act may affect the observable world by making the hearer bring the speaker another drink (a hot one) and may have an impact on the hearer's mental state by affecting his decision to not take the same drink. The utterance "the Moon is made of green cheese" is an assertive speech act where the speaker commits himself to the truthfulness of his statement. Humans utter different combinations of speech acts in response to previous speech acts advanced by the other participant in the dialogue. The exchange of such utterances according to a role-governed way (a protocol) constitutes a dialogue. Dialogue modeling is a subject of study in many domains like Linguistics, Natural Language Processing, Spoken Dialogue Systems and Multi-agent Systems, to name just a few.

In Multi-agent Systems, a formal dialogue model is a mathematical formalization that aims at capturing different aspects of natural dialogues. Dialogue may differ in nature with respect to certain criteria. (Walton and Krabbe, 1995, p. 66) have established a topology of dialogue types where they have classified dialogues with respect to the following criteria:

- **The initial situation of the dialogue:** it is the situation that proceeds the commencement of the dialogue.

- **The participant's individual goal:** the goal of each participant behind entering the dialogue.

- **The main goal of the dialogue:** it is what the dialogue is set to solve or to achieve. Note that the participants may have goals which are different from the main goal of the dialogue. One may get into a persuasion dialogue just to reveal the position of the other participant and not to convince him.

Three notable dialogues are well-studied in the literature, i.e. information-seeking, inquiry and persuasion.[1] In *information-seeking dialogues*, the initial situation is characterized by a personal ignorance where one participant lacks information and asks the other about it. In *inquiry dialogues*, the participants are found in a general ignorance about a proposition (or a hypothesis) and they collaborate to prove or disprove this proposition. In

[1] Other dialogues are also studied in the literature but they are out of the scope of the thesis (e.g. deliberation , negotiation, discovery, etc.).

TYPE	INITIAL SITUATION	PARTICIPANT'S GOAL	MAIN GOAL
Information-seeking	Personal ignorance	Gain, pass on, show, or hide personal knowledge	Spreading knowledge and revealing positions
Inquiry	General ignorance	Find a proof or destroy one	Growth of knowledge & agreement
Persuasion	Conflict of opinions	Persuade other party	Resolve or clarify issue
Explanation	Personal lack of understanding	Understand something	transfer of understanding

Table 5.1: The three dialogues from (Walton and Krabbe, 1995, p. 66). Note that Explanation dialogue does not figure in the topology.

persuasion dialogues, the initial situation is a conflict of opinion, the participants undertake the dialogue to persuade one another, and the main goal of the dialogue is the resolution of the conflict. Table 5.1 summarizes the major differences between these three dialogues.

The topology in general lacks an important dialogue type, i.e. explanation dialogues. In this type of dialogues one party tries to explain something to the other party. Recently, Walton has proposed a new dialogue that aims at capturing explanation dialogues in a series of papers (Walton, 2004, 2007, 2011, 2016). He has proposed two dialogue models, i.e. `CE` in (Walton, 2007) and `Explan` in (Walton, 2011, 2016).

The initial state of an explanation dialogue is the presence of an event, action or phenomenon which is accessible to both participants (factual). One of the participant (the explainee) cannot understand such fact, therefore he seeks understanding by asking for explanation(s) from the other participant (explainer) who is assumed to have a complete understanding of the fact in question. The main goal of the dialogue is to help the explainee to understand something he does not understand. To give an example, imagine a person A who has just arrived at a city. He gets out in a cloudy day and he sees that the sky is snowing but the snow is not accumulating on the ground, he asks another person B who sees the same thing and who happens to be a meteorologist. A requests an explanation from person B by saying "why it is snowing while there is no snowflakes on the ground?". After that,

B provides an explanation by saying *"Snow forms when the atmospheric temperature is at or below freezing. When snow reaches the ground and the ground temperature is above freezing, snowflakes will begin to melt. As a general rule, snow will not form if the ground temperature is 5 degrees Celsius"*.

As one may notice here this dialogue is authoritative. Person B has the authority over A. This makes it different from persuasion dialogues where authority is absent. Moreover, the information "it is snowing" and "the snow is not accumulating on the ground" are true and accessible to both parties. The subject of the dialogue is not whether it is accumulating or not but rather *why it is not accumulating?* Note here that the request for an explanation does not ask whether this is true or not, or what is the support or the justification of the statement. This differentiates explanation dialogues from persuasion and inquiry dialogues. In persuasion the goal is to persuade the other participant in an adversarial way. In inquiry the goal is proving or disproving such statement in a collaborative way.

Information-seeking dialogues are very similar to explanation dialogues. The difference is very subtle but rather crucial. In our interpretation, in an information-seeking dialogue the informee asks the informer some queries that demand *information*. So, what the informer says will have an impact on the informee *epistemic state* where he shifts from ignorance to non-ignorance. Moreover, once this exchange is realized the work of the informer is considered done. In an explanation dialogue, the explainer answers a query that asks for an explanation. Explanations convey information, however they convey more than a mere information, they convey understanding. Another subtle point is that at the time of requesting an explanation the explainee is not ignorant because he knows that it is snowing and the snow is not accumulating. In fact, he lacks understanding. So what the explainer says will have an impact on both, the epistemic as well as the understanding state of the explainee. Moreover, the work of the explainer does not end after the advancement of the explanation, instead he should make sure that the understanding has really taken place so that he can provide other explanations in case of a failure of understanding transference. Table 5.2 shows how the four dialogues differ from one another through an illustrating example.

In this chapter we build on the `Explan` dialogue model of explanation (Walton, 2011, 2016). We instantiate such dialogue model in our setting. As `Explan` is semi-formal, we provide a full formalization and extend it along different directions. This will result in a new dialogue model called `EDS`.

Information-seeking	Inquiry	Persuasion	Explanation
A does not know whether the snow has accumulated on the ground or not. B knows the information so A asks B if it is the case.	A and B are locked in a room and they cannot see whether the snow has accumulated or not but they may have some information that leverage the weather state. They collaborate to prove or disprove the hypothesis of "the snow has accumulated on the ground".	A believes that the snow has accumulated on the ground and B believes the contrary. Each of them try to persuade the other to adapt his position.	A and B know that the snow has not accumulated. B understands why it is the case and A does not. So, A asks B for an explanation.

Table 5.2: Examples showing the differences between the four types of dialogues.

5.2 Motivating Example

In what follows we give a dialogue example inspired from our application context. We have a knowledge base about Durum Wheat. We present a scenario where the `User` has queried the knowledge base about the subject of stubble breaking and then asks the `Reasoner` for an explanation about the content of the query.

Example 5.2.1. *Consider the query "Do we perform stubble breaking on the fields?" to which the* `Reasoner` *has answered yes. The following is an explanation dialogue about the subject of why do we perform stubble breaking?*

1. `User`*: Why do we perform stubble breaking?*
2. `Reasoner`*: The fields have some stubble standing on the soil. Also, they have residual plants which are left from previous crops. In order to be able to seed new crops we need to cut all stubble and remove residual plants.*
3. `User`*: Why the fields have some stubble on the soil?*
4. `Reasoner`*: After harvesting, some stubble stay incompletely cut.*
5. `User`*: I understand.*
6. `User`*: But I still don't understand why do we perform stubble breaking.*

7. `Reasoner`: *In fact without removing the stubble we cannot seed the next crop. Because the stubble will block the seeds from growing. Moreover, stubble breaking helps in creating a seedbed that will promote the germination of weed's seeds. Consequently, we will be able to remove these weed prior to the seeding of the new crop.*

8. `User`: *Ok, I understand now why do we perform stubble breaking.*

The dialogue starts at Stage (1) by an explanation request or a why-question about a factual statement addressed to the `Reasoner` by the `User`. At Stage (2) the `Reasoner` attempts to explain why do we perform stubble breaking. This attempt has evoked another explanation request at Stage (3) which the `Reasoner` attempted to explain at Stage (4). The `User` at Stage (5) declares that he has understood the last explanation. However, it seems that he is still unable to understand why do we perform stubble breaking (Stage 6). The `Reasoner` keeps trying to explain at Stage (7) where he provides another attempt to explain why do we perform stubble breaking. Fortunately, at Stage (8) the `User` finds this explanation intelligible. Consequently he acknowledges understanding and ends the dialogue.

As one should notice, this dialogue aims at making the `User` understand why the content of the query hold. He is not questioning the veracity of the query but rather asking why such thing is holding. Thus this dialogue has a pure explanatory purpose.

However, since the `Reasoner`'s knowledge base may have some inconsistencies the feedback from the `User` is very important. The `User` can challenge explanations when they seem implausible and possibly inconsistent with other facts. Imagine another course of action of Stage (7), let us refer to it as 7'. The `Reasoner` advances an explanation at Stage (7') and the `User` opposes to this explanation at Stage (8') as follows:

Example 5.2.2 (Alternative scenario)**.**

7'. `Reasoner`: *In fact stubble breaking is used to fight against fungal diseases. That is why we perform it.*

8'. `User`: *I do not think so because fungal diseases are caused by fungi which are microorganisms that are fought against by fungicides not by mechanical machines.*

9'. `Reasoner`: *I concede.*

10'. `User`: *I don't understand.*

The dialogue becomes argumentative where the `User` opposes to the `Reasoner` by giving an argument that attacks the `Reasoner`'s explanation. It is clear from the `User`'s position that the explanation contains a possibly imprecise information. The `Reasoner` concedes to this argument. This concession may express the fact that the `Reasoner` lacks sufficient knowledge to respond to the argument by being unable to recognize the argument (the argument's vocabulary is different from the one of the knowledge base) or by being unable to find a counterargument even if he could recognize the vocabulary. Whatever is the case it seems that the concession attitude of the `Reasoner` is not skeptical (he accepts what he is not able to counterattack). However, in our context we always assume that the `Reasoner` presents to the `User` a pre-computed set of arguments that he thinks that the `User` may hold. This type of argumentation dialogues has been recently studied in Hunter (2015) under the name of *asymmetric argumentation dialogues*.

Note that at the end the `User` can pursue the explanation dialogue and ask for further explanations. Here it seems that he fails to understand at Stage (10').

The dialogue respects certain rules and uses predefined speech acts like "why", "understand", etc. The dialogue also has a turntaking mechanism where the `User` and the `Reasoner` switch turns at each stage. In the next section we present Walton's dialogue model that attempts to capture such example.

5.3 Walton's Dialogue Model of Explanation

As said earlier, Walton has proposed two dialogue models of explanations. `CE` in (Walton, 2007) and `Explan` in (Walton, 2011, 2016). The two models follow the same general guidelines but they differ on certain aspects. `Explan` extends `CE` by adding a dialectical shift (change in dialogue) to examination dialogue Walton (2006) to test the understanding of the explainee as he can have fake understanding. Moreover, the `Explan` dialogue model extends `CE` by allowing a dialectical shift to an argumentation dialogue and gives the possibility to shift back.[2] The reason to do so is to give the user the possibility to evaluate explanations. We proceed by providing a brief introduction about `Explan`.

The system of explanation dialogues (denoted as `Explan`) is a two-player turntaking dialogue system of **explanation** (Walton, 2011). It takes place

[2]In fact this extension has been proposed first in our work in (Arioua and Croitoru, 2015) and later in (Walton, 2016).

between an *explainer* and an *explainee*. The speech acts of requesting and providing an explanation are represented as dialogue moves in the system.

The moves allowed within `Explan` are two distinct sets of moves: one for the explainer and another set for the explainee. The dialogue always starts with an assertion of a statement by the explainer, i.e. ASSERT(φ) and then the explainee requests an explanation for φ, i.e. EXPLAIN(φ)(φ is accessible by the two parties and believed to be true). Next, the explainer can offer an explanation attempt or declare her/his inability to explain. In the first case the explainee can ask for further explanations or acknowledge her/his understanding. If both parties are satisfied, the dialogue can be closed. If the explainee is not satisfied, he should ask further questions (QUESTION(φ)), continuing the dialogue until it has reached a point where either (a) he is satisfied or (b) his questioning must be closed off for practical reasons (Walton, 2011, p. 369). If the explainer is not satisfied, there should be a shift to an examination dialogue in which the explainee's understanding of the explanation is tested.

A comment about the speech acts ASSERT(φ) and QUESTION(φ) is in order. From the rules presented in Walton (2011, 2016) it is not clear how the questions are asked and how they are answered. In Table 5.3 we put for each speech act its appropriate reply. For ASSERT and QUESTION it is not clear in the model how they are replied to. However, the replies in the Table 5.3 have been confirmed through a personal communication with Douglas Walton.

Other aspects of the dialogue model are not less ambiguous, this would present a real challenge for an instantiation in a logical setting. Therefore, our first contribution is to fully formalize such dialogue model to avoid any ambiguities. The protocol of the `Explan` is simple and it is as Walton described *"..[it] is meant to be a simple and basic dialogue system specification on which specialized and more complex systems can be built.."* Walton (2011). Indeed, it cannot capture the exchange in Example 5.2.1 for the following reasons:

- Nested explanations request: the utterance at Stage (3) asks an explanation about another explanation. This is not accounted for in `Explan`.
- Turntaking: in Stage (5) and (6) the explainee (`User`) has taken two turns to make his point.
- Liberal protocol: in `Explan` the explainee and the explainer cannot backtrack to early stages. In the dialogue example the `User` has done

N	Speech acts	Descriptions	Replies
(1)	ASSERT(φ)	Putting forward a statement (reports an event or an action)	(2) or (3)
(2)	QUESTION(φ)	Asking whether it is the case that φ or not?	(1)
(3)	EXPLAIN(φ)	Requesting the explainer to explain φ	(4) or (5)
(4)	ATTEMPT(φ)	Responding to a previous explanation request that purports to convey understanding to the explainee	(6) or (7)
(5)	INABILITY(φ)	Conceding that the explainer has no explanation to offer at this point.	$\times$
(6)	POSITIVE(φ)	A response claiming that the explainee understands the explanation.	$\times$
(7)	NEGATIVE(φ)	A response claiming that the explainee does not understand the explanation	$\times$

Table 5.3: Speech acts in the `Explan` model. The replies refer to the entry number of the speech acts that reply to the one in question.

so in Stage (6) by responding to the explanation at Stage (2).

In the next section we propose the `EDS` (Explanatory Dialogue System) that extends and instantiates `Explan`.

5.4 The EDS Dialogue Model of Explanation

We describe here our dialogue model `EDS` of explanation (proposed in Arioua and Croitoru (2015)) which extends `Explan` and instantiates it in our logical setting. First let us define what is a dialogue system in general.

Definition 5.4.1 (Dialogue system). *A dialogue system is a tuple* $\mathcal{D}_{sys} = (\mathcal{P}r, \mathcal{C}, \mathfrak{R}, \mathcal{L}, \mathcal{K})$ *such that* $\mathcal{P}r = \{\mathtt{U}, \mathtt{R}\}$ *is the set of participants where* `U` *refers to* `User` *and* `R` *to* `Reasoner`*,* $\mathcal{C}$ *is a set of two disjoint finite sets* $\mathcal{C}_{\mathtt{U}}$ *and* $\mathcal{C}_{\mathtt{R}}$ *that denote* `U`*'s and* `R`*'s allowed locutions respectively,* $\mathfrak{R}$ *is an irreflexive binary relation defined over* $\mathcal{C}$ *called the reply relation,* $\mathcal{L}$ *is a logical language called the content language and* $\mathcal{K} \subseteq \mathcal{L}$ *is the background knowledge base which is accessible by both participants.*

This dialogue system describes the general components of an asymmetric arbitrary dialogue with two participants. Asymmetric dialogues are dialogues where the set of locutions for each participant is different from the other's set (may overlap). This formalization is close to the common formalization of argumentation dialogues used in Prakken (2009); Amgoud et al.

(2006). The main difference is that almost all argumentation dialogues studied in the literature except for Hunter (2015) are symmetric. In what follows we instantiate these components to describe a dialogue system called *the explanatory dialogue system* (`EDS`).

Participants. In a general dialogue system the participants refer to two agents (sometimes more) with possibly different nature, they may be human agents or mixed agents (e.g. machine with human) where each participant plays a role. In the explanatory dialogue system `EDS` we are interested in the case of machine-human agents where the machine is called the `Reasoner` and denoted as `R`, while the human is called `User` and denoted as `U`. `U` plays the role of the *explainee* and the other plays the role of an *explainer*.

Topic and the content language. Following Walton (2011, 2016); Arioua and Croitoru (2015), given a factual information F which can be drawn from the background knowledge base, the topic of any dialogue of `EDS` is a discussion that aims to get `U` to understand why F is the case. In this dialogue system, `R` tries to provide explanations to `U`. The content language defines the language by which the participants communicate. Here we consider the logical language $\mathcal{L}$ of existential rules previously seen in Chapter 3, Section 3.2.

Background knowledge. The background knowledge is the knowledge mainly held by the `Reasoner` and accessible to the `User`. It is denoted as $\mathcal{K} = (\mathcal{F}, \mathcal{R}, \mathcal{N})$ and it contains a set of facts, set of rules and set of constraints. In `Explan`, it is not clear whether a knowledge base is used or not but it seems that in Walton (2011) the background knowledge is referred to as the context but without any formal definition.

An important aspect of dialogue systems is the protocol which regulates the exchange of utterances between the participants (McBurney and Parsons, 2009). The protocol's syntax handles the syntactical validity of the utterances regardless of their content. The protocol's semantics handles the meaning of the utterances and its impact on their validity. In Subsection 5.4.1 we start by presenting the protocol's syntax of `EDS`. Next, in Subsection 5.4.2 we present the protocol's semantics. In the literature, two crucial subjects are always handled separately from the semantics although being completely related to it, are commitments and termination. We dedicate two sections to discuss them separately for their importance. We discuss the integration of commitment and understanding stores and study their outcomes in Subsection 5.4.3 and we discuss the conditions of termination and success in Subsection 5.4.4.

5.4.1 Protocol's syntax

We follow McBurney et al. (2002) and present the syntax independently from the semantics. The basic building of the syntax is the set of allowed locutions (speech acts). We distinguish `EDS` from other dialogue systems [3] by the following locutions:

- EXPLAIN: `U` requests an explanation for a fact.
- ATTEMPT: `R` gives a response (to a previous explanation request) that attempts to provide an explanation.
- POSITIVE: `U` claims that he understands the explanation.
- NEGATIVE: `U` claims that he does not understand the explanation.
- INABILITY: `R` declares inability to explain. Or as in Walton's words `R` utters: "I can't explain it".

Formally:

Definition 5.4.2 (Locutions). *Give a dialogue system $\mathcal{D}_{sys} = (\mathcal{P}r, \mathcal{C}, \mathfrak{R}, \mathcal{L}, \mathcal{K})$ as instantiated previously. The set of all allowed locutions for each participant is presented as follows:*

- $\mathcal{C}_{\mathtt{R}} = \{\text{ATTEMPT}, \text{INABILITY}\}$, *called* `R`*'s allowed locutions.*
- $\mathcal{C}_{\mathtt{U}} = \{\text{EXPLAIN}, \text{POSITIVE}, \text{NEGATIVE}\}$, *called* `U`*'s allowed locutions.*

When there is no risk of ambiguity, we may slightly abuse notation and refer to the set of allowed locutions $\mathcal{C}$ as the union of its subsets.

As one can see, the set of locutions is partitioned with respect to the participants. The difference between the sets is due to the asymmetry of roles where the `Reasoner` plays the role of an explainer and the `User` plays the role of an explainee. This asymmetry is in fact due to the authoritative nature of explanation dialogues where the explainer is assumed to possess knowledge which the explainee wants to acquire in order to understand certain statements. An example of the asymmetry is that the `User` is the one

[3] For instance, `Explan`, the dialogue systems of argumentation CB Walton (1984), DC Mackenzie (1979), PPD Walton and Krabbe (1995), to name a few.

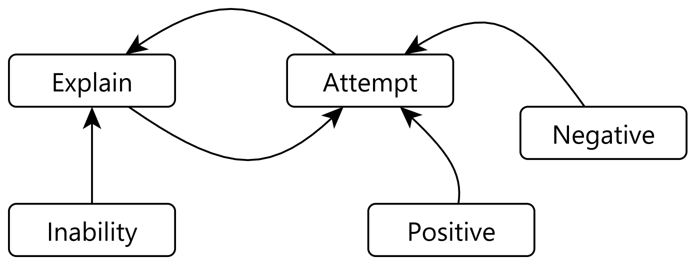


Figure 5.1: The reply relation $\mathfrak{R}$ over the locutions. The direction of the arrows are read as "replies to".

who asks for an explanation (EXPLAIN) and the `Reasoner` is the one who provides the explanation (ATTEMPT) not the other way around. We should highlight that in `EDS` the locutions ASSERT and QUESTION are dropped. As our logical setting offers querying facilities, QUESTION() and ASSERT() are dispensable as the need for them can be fulfilled any time through the querying engine. Therefore, we see no practical need to incorporate them.

Reply relation. The reply relation $\mathfrak{R}$ in Figure 5.1 specifies which locution replies to which. As it is indicated, the EXPLAIN request locution is either replied to by an ATTEMPT that provides an explanation, or by a declaration of inability INABILITY. ATTEMPT is replied to by an explanation request EXPLAIN which asks for a further explanation of some parts of the first explanation. Or by NEGATIVE to disacknowledge understanding or by POSITIVE to acknowledge understanding. Note that the EXPLAIN reply to ATTEMPT is a feature of `EDS` and is not presented in `Explan`. The usefulness of such extension resides in allowing nested explanation requests.

Uttering these locutions in a dialogical context constitutes an explanation dialogue. Generally, a dialogue is a sequence of **utterances** between two parties (or more). We follow Atkinson et al. (2005) and we represent utterances by a two-layer syntax: the wrapper layer and the content layer. The wrapper layer encompasses locutions (e.g. EXPLAIN, ATTEMPT, etc) which represent the illocutionary force of the inner content. The content layer includes the following components: the identifier x of the speaker, the `identifier` i of the utterance, the `target` t of the utterance and the content A of the utterance expressed in the content logical language $\mathcal{L}$. It is defined formally as follows:

Definition 5.4.3 (Utterance)**.** *Let $\mathcal{C}$ be the set of all allowed locutions. An utterance u has the form* $\mathrm{X}(x, i, t, A)$ *such that:*

1. $i \geq 1$.
2. $t < i$.
3. $x \in \mathcal{P}r$.
4. $X \in \mathcal{C}_x$.
5. *A are well-formed formulas of $\mathcal{L}$.*

We use the notations $\texttt{loc}(u) = \mathrm{X}$*,* $\texttt{part}(u) = x$*,* $\texttt{id}(u) = i$*,* $\texttt{target}(u) = t$ *and* $\texttt{Content}(u) = A$ *to denote respectively the locution, the participant, the identifier, the target and the content of the utterance u. We may sometimes use the notation* $\mathrm{X}(.)$ *and ignore the parameters to denote an anonymous utterance (e.g.* EXPLAIN$(.)$*). We say the utterance u is* ***equivalent*** *to another utterance u' and we write $u = u'$ if and only if* $\texttt{loc}(u) = \texttt{loc}(u')$, $\texttt{part}(u) = \texttt{part}(u')$*,* $\texttt{target}(u) = \texttt{target}(u')$ *and* $\texttt{Content}(u) = \texttt{Content}(u')$*.*

The definition imposes that the identifier of the utterance should be greater than 0 (Clause 1) and the target's identifier should be smaller than the identifier of the utterance itself (Clause 2). This means that the identifier can be seen as the *timestamp* of the utterance that indicates when it was uttered. The target should be smaller then the identifier since the utterance replies to a previous utterance. In Clause 3 the speaker should be *known*, i.e. it should be in the set of participants $\mathcal{P}r$. In Clause 4 the the locution of any utterance has to be in the set of the speaker's allowed locutions. Finally, Clause 4 stipulates that the content should be well-formed with respect to the content language $\mathcal{L}$. An utterance is equivalent to another if it has the same locution, participant, target and content.

A comment about the parameter `part` is in order. It is legitimate to say that we do not need this parameter in the utterance as the locutions can unambiguously tell the participant. Indeed, this is correct. However, when we extend this framework to incorporate symmetric argumentation dialogues, as we shall see in Section 5.5, we will need this parameter to differentiate between utterances.

After defining the most important part of a dialogue, i.e. utterances, we define a dialogue as a sequence of utterances.

Definition 5.4.4 (Explanation dialogue)**.** *An explanation dialogue (dialogue for short) d is a possibly infinite sequence $d = (u_1, \ldots, u_n, \ldots)$ of utterances u_i where $i > 0$ and for all $u_i \in d$,* $\texttt{id}(u_i) = i$*.*

The finite sequence $(u_1, \ldots, u_n)$ *, denoted as* d_n*, is the prefix of the sequence* d *where* n *is the number of utterances in* d_n *and* u_n *is the most recent utterance.* d_n *is referred to as the dialogue at stage* n*.* d_0 *is called the empty dialogue. The set of all possible dialogues is denoted by* $\mathcal{D}^\infty$*. We denote by* $d_i \cdot d_j$*, and* $d_i \cdot u$ *the concatenation of the dialogues* d_i *and* d_j *and the dialogue* d_i *with the utterance* u *respectively.*

Let us take an example of a dialogue.

Example 5.4.1 (Dialogue)**.** *Consider Example 5.2. Let* A *and* B *be well-formed formulas of the language* $\mathcal{L}$*. As the semantics is not yet considered we do not give the logical form of* A *and* B*, we present only their natural representation to show the intuition.*

- A : *we perform stubble breaking?*
- B : *The fields have some stubble standing on the soil. Also, they have residual plants which are left from previous crops. In order to be able to seed new crops we need to cut all stubble and remove residual plants.*

Consider the following dialogue of 4 steps:

$$
\begin{aligned}
d_4 = (&\\
&\textsc{explain}(\mathtt{U}, 1, 0, A),\\
&\textsc{attempt}(\mathtt{R}, 2, 1, B),\\
&\textsc{negative}(\mathtt{U}, 3, 2, B),\\
&\textsc{positive}(\mathtt{U}, 4, 2, B)\\
&)
\end{aligned}
$$

This is a dialogue where `U` *asks for an explanation. Next,* `R` *fulfills the request. After that,* `U` *disacknowledges understanding then after a while he/she acknowledges understanding.*

The parameters of the utterance contextualize the intention of uttering its locution. The locution ATTEMPT would make no sense if we do not specify to which locution is replying and what is the content of the reply. For instance, the locution ATTEMPT is only an act of expressing the intention to make an attempt of explanation (a speech act according to (Austin, 1975)) it conveys nothing beyond that. Table 5.4 indicates for each locution its complete parameters alongside to its interpretation.

Utterance	Interpretation
$\textsc{explain}(\mathtt{U}, i, t, A)$	U responds to the utterance u_t by requesting an explanation of A. If $t = 0$, then this utterance starts the dialogue.
$\textsc{attempt}(\mathtt{R}, i, t, B)$	R responds to the utterance u_t by an explanation B.
$\textsc{negative}(\mathtt{U}, i, t, B)$	U has not understood the explanation B presented in the utterance u_t.
$\textsc{positive}(\mathtt{U}, i, t, B)$	U has understood the explanation B presented in the utterance u_t.
$\textsc{inability}(\mathtt{R}, i, t, A)$	R is unable to provide an explanation for the explanation request of the utterance u_t.

Table 5.4: Interpretation of utterances.

Example 5.4.2 (Cont'd Example 5.4.1)**.** *The utterance* $\textsc{negative}(\mathtt{U}, 3, 2, B)$ *means that* U *at Stage 3 has responded to the utterance at Stage* 2 *by the content* B*. The utterance right after, i.e.* $\textsc{positive}(\mathtt{U}, 4, 2, B)$*, means that* U *at stage 4 has responded to the utterance at Stage* 2 *by the content* B*.*

Please note that we are studying here the syntactical aspects of the dialogue. The real meaning of each utterance depends completely on meaning of its content. We shall study this in the semantics (Subsection 5.4.2).

As in the motivating dialogue Example 5.2.1, R and U take turns. The most basic form of turntaking is *unique-move* turntaking which is used in `Explan`. This turntaking gives the participants the possibility to advance one utterance then hand out the turn to the other to respond. A more liberal turntaking is the *multiple-move* turntaking where the turn shifts after several utterances (which is the case for the example). We follow the idea of *liberal argumentation dialogues* of Prakken (2005) and we choose the multiple-move turntaking for its generality. Sure, this turntaking makes the dialogue difficult to handle computationally, however it offers a natural correspondence with day-to-day explanation dialogues where one uses sufficient utterances to express his point. In addition such turntaking allows the participants to interact in a sophisticated manner to capture complex debates.

Definition 5.4.5 (Turntaking function)**.** *A turntaking function* $\mathcal{T}$ *is defined over the set of all possible dialogues as follows:* $\mathcal{T} : \mathcal{D}^{\infty} \longrightarrow 2^{\{\mathtt{U},\mathtt{R}\}}$*.* $\mathcal{T}$ *assigns* *to every dialogue the next legal turn as follows:*

- $\mathcal{T}(d_0) = \{\mathtt{U}\}$, $\mathcal{T}(d_1) = \{\mathtt{R}\}$, $\mathcal{T}(d_i) = \{\mathtt{U}, \mathtt{R}\}$, $\forall i > 1$.

Depending on the previous utterances within a dialogue, the next utterance to be put forward can be legal or illegal. It depends whether it replies properly to previous utterances.

Definition 5.4.6 (Legal reply)**.** *Given a finite dialogue $d_n = (u_1, u_2, \ldots, u_n)$ at stage n and two utterances u_i and u_j in d_n. We say that u_i replies **legally** to u_j in d_n if and only if :*

(1) $(\mathtt{loc}(u_j), \mathtt{loc}(u_i)) \in \mathfrak{R}$.

(2) $\mathtt{part}(u_i) \in \mathcal{T}(d_{i-1})$.

(3) $\mathtt{target}(u_i) = \mathtt{id}(u_j) = j$.

(4) $\mathtt{part}(u_i) \neq \mathtt{part}(u_j)$.

If non of the previous conditions are met then the utterance u_i is an illegal reply to u_j in d_n. Recall that $(\mathrm{X}, \mathrm{X}') \in \mathfrak{R}$ means that the locution X' replies to X.

For an utterance u_i to be a legal reply to another utterance u_j the clauses impose that the locution of u_i should be a correct reply to the locution of u_j with respect to the reply relation $\mathfrak{R}$, and it is the turn of the participant to speak. Moreover, they impose that the target utterance u_j should have already been uttered. Finally it dictates that the $\mathtt{part}(u_i)$ should not be replying to himself.

Example 5.4.3 (Legal reply)**.** *Consider the following dialogue:*

$d_4 = ($

$\textsc{explain}(\mathtt{U}, 1, 0, A)$,

$\textsc{attempt}(\mathtt{R}, 2, 1, B)$,

$\textsc{positive}(\mathtt{U}, 3, 1, B)$,

$\textsc{positive}(\mathtt{U}, 4, 3, B)$

$)$

The reply $\textsc{positive}(\mathtt{U}, 3, 1, B)$ *to* $\textsc{explain}(\mathtt{U}, 1, 0, A)$ *is illegal because the tuple* $(\textsc{explain}, \textsc{positive})$ *is not in* $\mathfrak{R}$ *as opposed to the reply* $\textsc{attempt}(\mathtt{R}, 2, 1, B)$ *to* $\textsc{explain}(\mathtt{U}, 1, 0, A)$ *which is legal. Notice also that, the reply* $\textsc{positive}(\mathtt{U}, 4, 3, B)$ *is illegal because* R *is replying to himself in addition to the fact that* $(\textsc{positive}, \textsc{positive}) \notin \mathfrak{R}$. *The following dialogue is free of illegal replies.*

$$d'_3 = ($$
$$\textsc{explain}(\mathtt{U}, 1, 0, A),$$
$$\textsc{attempt}(\mathtt{R}, 2, 1, B),$$
$$\textsc{attempt}(\mathtt{R}, 3, 1, B)$$
$$)$$

As one may notice, the reply of $\textsc{attempt}(\mathtt{R}, 3, 1, B)$ to $\textsc{explain}(\mathtt{U}, 1, 0, A)$ is considered legal with respect to the conditions seen before. The problem is that this utterance is a duplicate of the reply $\textsc{attempt}(\mathtt{R}, 2, 1, B)$ to $\textsc{explain}(\mathtt{U}, 1, 0, A)$. In fact reply's validity is a concept with a limited scope, it is only concerned with a local context, i.e. between two utterances. In what follows we capture the global context of dialogues and we introduce *the syntactical validity of a dialogue.*

Definition 5.4.7 (Syntactically valid dialogues)**.** *Given a finite dialogue d_n at stage n. A dialogue d_n is syntactically valid if it respects the following rules:*

1. ***Empty dialogue rule*** *($n = 0$):*

   (R_1) d_0 is syntactically valid by convention.

2. ***Commencement rule*** *($n = 1$):*

   (R_2) $d_1 = (u_1)$ is syntactically valid iff $u_1 = \textsc{explain}(\mathtt{U}, 1, 0, A)$.

3. ***Dialogue rule*** *($n \geqslant 2$):*

   (R_3) d_{n-1} should be syntactically valid and the reply u_n to u_j where $j = \mathtt{target}(u_n)$ is legal and there is no utterance u_i, $i < n$ such that $\mathtt{target}(u_i) = \mathtt{target}(u_n)$ and $u_i = u_n$.

The definition indicates that an empty dialogue is a syntactically valid dialogue (R_1). Furthermore, a syntactically valid dialogue always starts with an explanation request made by U (R_2). It also imposes that for a dialogue to be syntactically valid all the replies are syntactically valid and no reply is repeated (R_3).

Another important feature of the dialogue is that we can reply to early utterances. This breaks the sequentiality of the dialogue. In Prakken (2009) these types of protocols are called *multiple-reply* protocols.

So far we have defined the syntactical part of the protocol. In the next section we shift to the semantics part.

5.4.2 Protocol's semantics

We address the semantics aspect of the dialogue where we are concerned with meaning of the content of the utterances and the meaning behind a sequence of utterances. For instance, the utterance $\text{ATTEMPT}(\mathtt{R}, 1, 0, E)$ conforms Definition 5.4.3, however it would not be semantically legal if E were not to be a semantically legal explanation. Following the same structural organization of the previous section, we start by defining the semantic legality of utterances and replies, then we define the semantic validity of dialogues.

To define the semantic legality of an utterance or a reply we need first to define what is an explanation. As the subject of defining explanatory models is very controversial on many aspects, e.g. philosophical, logical, psychological, etc. (Pitt, 1988). In (Arioua and Croitoru, 2015) we have proposed an abstract setting where an abstract explanatory model is defined as a tuple $\mathbf{E} = (\mathcal{L}, \rightarrowtail)$[4] which consists of the content language $\mathcal{L}$ and an explanatory relation denoted as $\rightarrowtail$ which is defined over $2^{\mathcal{L}} \times 2^{\mathcal{L}'}$ such that $\mathcal{L}' \subseteq \mathcal{L}$ is the set of well-formed formulas that correspond to facts. $\rightarrowtail$ relates those well-formed formulas (wffs) in $\mathcal{L}$ that can be considered as an *explanation* to the formula to be explained called the *explanandum*. The explanandum should be factual. An explanation is composed of the explanans which are the formulas which together *bear* explanatory relevance to the explanandum (Figure 5.2). Formally, given a set E of wffs and a fact F we read $E \rightarrowtail F$ as "E is an explanation of F". Note that no constraints are imposed on the explanatory relation. Certainly, a fine-grained axiomatization is needed which unfortunately falls out of the scope of this thesis.

As far as our logical context is concerned many logical frameworks that account for explanation can be found in the literature (*Axiom pinpointing*; (Schlobach et al., 2003), *Justification Oriented Proofs* (Horridge et al., 2010), *Causes* (Meliou et al., 2014)). The best-known is the framework of *Logic-based Abduction* (Eiter and Gottlob, 1995). This framework has been

[4]Please note that this definition is slightly different from the one in Arioua and Croitoru (2015). However, this change does not impact the soundness of the dialogue system.

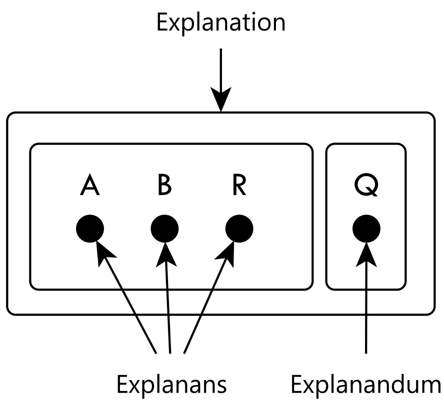


Figure 5.2: Explanation according to Deductive-Nomological Model of Hempel (1965).

applied in many domains which are far to be cited exhaustively here. There are works on formalizing Logic-based Abduction in setting similar to ours like (Calvanese et al., 2013; Du et al., 2014). One could easily instantiate the explanatory relation using these works on our logical setting if one chooses such framework for generating explanations. However, as our goal in this chapter is not to study which explanation is better we consider a simplistic instantiation of the explanatory relation. In this instantiation we define the explanatory relation as an inference relation such that the explanation minimally entails the explanandum. In addition, we impose that the explanation and the explanandum are consistent together (a similar definition can be found in (Falappa et al., 2002)).[5]

Definition 5.4.8 (Explanation)**.** *Given a knowledge base $\mathcal{K} = (\mathcal{F}, \mathcal{R}, \mathcal{N})$ and a fact $\mathcal{Q}$. An explanation $E \subseteq \mathcal{F} \cup \mathcal{R}$ of $\mathcal{Q}$ in $\mathcal{K}$ is a set of facts and rules such that:*

1. *$E, \mathcal{N} \not\models \bot$ (consistency).*

2. *$E \models \mathcal{Q}$ (entailment).*

3. *$\nexists E' \subseteq E$ such that (1) and (2) are verified (minimality).*

[5]The symbol $\rightarrowtail$ is borrowed from their work.

One may be tempted to believe that an explanation in this sense is an argument in the sense of Definition 3.3.1, page 50. This is simply not the case, because an argument does not contain rules in our definition. Nevertheless, it is indeed a deductive argument in the sense of Besnard and Hunter (2008).

Notation 5.4.1. There may be more than one explanation for a given fact, thus we denote by $\mathcal{E}xp(F)$ the set all explanations of F. For an explanation E we denote by $\texttt{Rules}(E)$ and $\texttt{Facts}(E)$ the set of rules and facts used in the explanation respectively.

It is not hard to conclude that an explanation should have always some facts in it. Because the rules need some facts on which they will be applied to derive the explanandum. In fact this is straightforwardly intuitive because ordinary or scientific explanations need to start from initial conditions which are taken to be factual.

Property 2 (Factuality). Given an explanation E of $\mathcal{Q}$. $\texttt{Facts}(E) \neq \emptyset$.

Let us take an illustrative example of explanations.

Example 5.4.4. *Imagine that we have a knowledge base about employees and their salaries.* $\mathcal{Q} = has\_salary(Tom, x)$ *is a fact that says that Tom has a salary. The salary is unknown but it is to be taken as a fact that he has a salary. Now the* User *can ask for an explanation "Why Tom has a salary" or "Why is it the case that Tom has a salary". An explanation* E_1 *of* F *is as follows:*

$$
\begin{aligned}
E_1 = \{ & \\
& works\_at(Tom, UM), \\
& university(UM), \\
& works\_at(x, y) \wedge university(y) \rightarrow has\_salary(x, z) \\
& \}
\end{aligned}
$$

Where $\texttt{Facts}(E) = \{works\_at(Tom, UM), university(UM)\}$ *and* $\texttt{Rules}(E) = \{works\_at(x, y) \wedge university(y) \rightarrow has\_salary(x, z)\}$. *Below another explanation* E_2*:*

$$
\begin{aligned}
E_2 = \{ & \\
& retired\_from(Tom, UM), \\
& university(UM),
\end{aligned}
$$

$retired\_from(x, y) \wedge university(y) \rightarrow has\_salary(x, z)$
}

Note that the following:

$E_3 = \{$
$works\_at(Tom, UM),$
$university(UM),$
$student(Ahmed),$
$works\_at(x, y) \wedge university(y) \rightarrow has\_salary(x, z)$
$\}$

is not an explanation because $student(Ahmed)$ is irrelevant which makes E_3 not minimal. Note also that, $E_4 = E_1 \setminus \{works\_at(Tom, UM)\}$ is not an explanation because it violates entailment (Definition 5.4.8, Clause 2). Another invalid explanation which violates consistency is as follows:

$$E_5 = E_1 \cup \{retired\_from(Tom, UM)\}$$

A comment is in order here about consistency. The fact that E_1 and E_2 dictate possibly inconsistent facts, i.e. $retired\_from(Tom, UM)$ and $works\_at(Tom, UM)$, does not affect the fact that they are correct explanations. It only says that they are competitive and the background knowledge base $\mathcal{K}$ is possibly inconsistent.

Utterances	Replies	Semantics legality
EXPLAIN$(\mathtt{U}, i, t, F)$	ATTEMPT$(\mathtt{R}, i+1, i, E)$	$\mathcal{K} \models F$ and $E \rightarrowtail F$.
EXPLAIN$(\mathtt{U}, i, t, F)$	INABILITY$(\mathtt{R}, i+1, i, F')$	$\mathcal{K} \models F$, $F = F'$ and $\nexists E$ such that $E \rightarrowtail F$.
ATTEMPT$(\mathtt{R}, i, t, E)$	EXPLAIN$(\mathtt{U}, i+1, i, F)$	$\mathcal{K} \models F$ and $F \subseteq \mathtt{Facts}(E)$.
ATTEMPT$(\mathtt{R}, i, t, E)$	POSITIVE$(\mathtt{U}, i+1, i, E')$	$E = E'$.
ATTEMPT$(\mathtt{R}, i, t, E)$	NEGATIVE$(\mathtt{U}, i+1, i, E')$	$E = E'$.

Table 5.5: The replies and their semantic conditions. $\mathcal{K}$ is the background knowledge, F is a fact, E is an explanation and i, t are natural numbers.

For a dialogue to be semantically valid its utterances and replies should be semantically legal. In Table 5.5 we present the conditions under which a given utterance and their replies are considered semantically valid.

Definition 5.4.9 (Semantically valid dialogues)**.** *Let d_n be a syntactically valid dialogue. The dialogue d_n is semantically valid if and only if all utterances $u \in d_n$ and every reply u to u' in d_n is semantically legal (cf. Table 5.5). Note that the empty dialogue d_0 is semantically valid.*

Many questions arise, for instance when does the dialogue is considered successful? and when does it terminate? Moreover, how we maintain the coherence of dialogue to avoid circular explanations. In the next section we answer these questions.

5.4.3 Commitment and understanding stores

Commitments are set of statements to which a participant in a dialogue is committed. When a person x says in a dialogue that "I claim that the Moon is made of green cheese" he becomes instantaneously committed to the fact that "the Moon is made of green cheese", he may (or may not) believe in its truthfulness but he claims that it is true. The notion of commitments differs from a dialogue system to another. In the literature, precisely in (Hamblin, 1970), commitments are treated dialectically. In Hamblin's words *"A speaker who is obliged to maintain consistency needs to keep a store of statements representing his previous commitments, and require of each new statement he makes that it may be added without inconsistency to this store."* (Hamblin, 1970, p. 257). However, others (precisely (Walton and Krabbe, 1995)) give a larger interpretation of commitments by considering them as obligations to perform actions. For instance, a participant is obliged to provide a counterattack when his asserted proposition is challenged. It seems that this view englobes the one of Hamblin. Walton and Krabb's approach to commitments is general as it handles, for instance, social actions in negotiation dialogues (a magazine x commits to ship the item A to a customer y). However, in other dialogues systems, such as (Prakken, 2005; Parsons et al., 2003), only dialectical commitments in the sense of Hamblin are considered. In fact the consistency requirement is dropped when commitments do not regulate the advancement of utterances within the dialogue. In other words, the speaker is not obliged to maintain the consistency of his commitments as he proceeds in the dialogue.

For the dialogue system `Explan`, both participants are assumed to have commitment stores that keep track of their commitments. But it is not clear why such commitments are used or how they are updated. In addition, given the asymmetry of the dialogue system it is not clear why the explainee (`User` in our case) should be consistent. One should be able to ask (separately) for an explanation of two completely inconsistent facts and he is still eligible to get an explanation and there is no reason to deprive him from that. In Arioua and Croitoru (2015) we proposed to equip *only* the `Reasoner` with commitments store because commitments of the `User` have no impact on the coherence of the dialogue. The `Reasoner`'s commitments store is particularly needed to ensure that the `Reasoner` gives consistent explanations. Therefore, the `Reasoner`'s commitments store indeed constraints his advancement of utterances. Although being useful in certain cases, this requirement should be dropped for the following reason. Recall that in Example 5.4.4 we have seen that explanations are self-consistent but there could be some competing explanations which cannot be held together. If we follow Arioua and Croitoru (2015) the `Reasoner` would not be able to present the two explanations within the same dialogue, which could be qualified as an incomplete and unfaithful way of explaining. However, we retain from Arioua and Croitoru (2015) the use of dialectical shifts (change of dialogue's type) to argumentation dialogues to evaluate these competing explanation. In retaining such feature, we are obliged to equip the `User` with a commitment store because this will become important in the argumentation dialogue where the commitments may be used in determining the winner and termination.

To sum up, in `EDS` each participant has a commitment store that indicates at each moment a set of formulas to which the participant is committed.

Definition 5.4.10 (Commitment store)**.** *Given a dialogue $d_n = (u_1, \ldots, u_n)$. A commitment store $\mathcal{CS}_x^i \subseteq \mathcal{L}$ is a set of formulas from $\mathcal{L}$. We denote by $\mathcal{CS}_x^i$ such that $x \in \{\mathtt{U}, \mathtt{R}\}$ the commitment store of the participant x after uttering u_i in d_n. Given a background knowledge base $\mathcal{K} = (\mathcal{F}, \mathcal{R}, \mathcal{N})$, we say that the commitment store $\mathcal{CS}_x^i$ is inconsistent if and only if $C\ell^*_{\mathcal{R}}(\mathcal{CS}_x^i) \models \bot$.*

Commitments are used to solve the problem of maintaining the consistency. However, there is another problem we may encounter in explanation dialogues which is *circular explanations.* They appear when the `Reasoner` tries to explain a fact with an explanation that contains some parts which are not yet understood by the `User`. This is in fact a form of circular reasoning which is often ascribed to the fallacy of begging the question that is studied in argumentation dialogues (Mackenzie, 1979). In Walton (2004), where Walton informally described his dialectical model of explanation, he proposed to build on the notion of Hamblin's commitment store to capture understanding.[6] He later brought up the idea in the `CE` dialogue model of

[6] "*To define understanding in dialogue, we need to build on Hamblin's notion of the commitment store of a participant in a dialogue.*" (Walton, 2004, p. 7)

explanation (Walton, 2007) where he highlighted the point of using some sort of *understanding states* to represent the understanding of each participant (Walton, 2007, Table. 2). To be more precise, the problem of circular explanations was not the motivation of capturing understanding in `CE` it was rather the question whether understanding has been really transferred or not.

So in what follows we develop Walton's first intuition by proposing an *understanding store* which is exclusively attributed to the `User`. *The understanding store* serves as an understanding indicator of the `User`'s current understanding state and it is used to avoid circular explanations.

Definition 5.4.11 (Understanding store). *Given a dialogue d_n. An understanding store $\mathcal{US}^i \subseteq \mathcal{L}$ is a set of formulas from $\mathcal{L}$. i refers to the content of the understanding store after uttering u_i in d_n.*

We do not use any participant index in the understanding store as it is clear to whom it is attributed (to the `User`). Some important comments are worth highlighting about understanding stores.

- if $\mathcal{US}$ does not contain F then the `User` understands F.
- $\mathcal{US}$ may or may not be consistent.

For `CE` and `Explan`, the state of not understanding F is denoted as not-F. However, not-F is open to many interpretations. One could say that it means that the `User` understands the negation of F but not F. To avoid such confusion, we take $\mathcal{US}$ as representing what is not yet understood instead of what has been understood. In other words, if there is something which is not understood by the `User` then it should be in the understanding store and every thing that is not in the understanding store is assumed to be understood by the `User`. The state of understanding of the `User` evolves over time. In the beginning of the dialogue the understanding store is *conventionally* assumed to be empty.[7] Then in function of the utterances advanced by both parties the store is altered. The same thing happens for commitment stores. The rules that organize how stores are modified are called in the literature *Effect rules*.

Definition 5.4.12 (Effect rules). *Let $d_n = (u_1, \ldots, u_n)$ be a dialogue and let $\mathcal{CS}^n_{\mathtt{R}}$, $\mathcal{CS}^n_{\mathtt{U}}$ and $\mathcal{US}^n$ be, respectively, the commitment stores of* `R` *and* `U` *and the understanding store after advancing the utterance u_n.*

[7] In fact there is no theoretical reason to impose that one should start with an empty understanding store. It is for practical reasons and convenience that we assume so.

1. *If* $u_n = \text{EXPLAIN}(\mathtt{U}, i, t, F)$ *then:*
   (a) $\mathcal{US}^n = \mathcal{US}^{n-1} \cup \{F\}$.
   (b) $\mathcal{CS}^n_{\mathtt{R}} = \mathcal{CS}^{n-1}_{\mathtt{R}} \cup \{F\}$.
   (c) $\mathcal{CS}^n_{\mathtt{U}} = \mathcal{CS}^{n-1}_{\mathtt{U}} \cup \{F\}$.
2. *If* $u_n = \text{ATTEMPT}(\mathtt{R}, i, t, E)$ *then* $\mathcal{CS}^n_{\mathtt{R}} = \mathcal{CS}^{n-1}_{\mathtt{R}} \cup \{E\}$.
3. *If* $u_n = \text{POSITIVE}(\mathtt{U}, i, t, E)$ *then:*
   (a) $\mathcal{US}^n = \mathcal{US}^{n-1} \setminus \{F'\}$ *such that* $F' = \mathtt{Content}(u_j)$ *where* $j = \mathtt{target}(u_t)$.
   (b) $\mathcal{CS}^n_{\mathtt{U}} = \mathcal{CS}^{n-1}_{\mathtt{U}} \cup \{E\}$.

Utterances with the locutions INABILITY *and* NEGATIVE *have no effect on the stores.*

When the `User` requests an explanation we add the explanandum F to his understanding store (1.a). The `User` and the `Reasoner` become committed to F (1.b and 1.c) because it is agreed upon in explanation dialogues that the explanandum F should be held true prior to requesting its explanation. When the `Reasoner` provides an explanation he intuitively becomes committed to it (2). If the `User` acknowledges understanding in response to an explanation E, then we revoke the explanandum of the explanation from $\mathcal{US}$ (3.a). This means that the `User` declares that he could understand the explanandum thanks to the explanation. At his point, he becomes committed to the truthfulness of the explanation E. Since he acknowledges understanding then he is automatically committed to it.

As we noted earlier, the commitment stores will not be used in regulating the advancement of utterances. However, the understanding store is used to avoid circular explanations. In what follows we extend the semantic validity of Definition 5.4.9 by the following rule.

Definition 5.4.13 (Closing semantic validity)**.** *Let d_n be a syntactically valid dialogue. The dialogue d_n is semantically valid if and only if for all u_i in d_n, $i \leqslant n$ if* $\mathtt{loc}(u_i) = \text{ATTEMPT}$ *then* $F \cap \mathcal{US}^i = \emptyset$ *such that* $F = \mathtt{Content}(u_i)$.

In this rule we simply forbid the `Reasoner` from advancing an explanation that contains something which has been declared as “not understood” by the `User`.

5.4.4 Termination and success

In almost any dialogue system the termination condition should be specified. The conventional termination criterion is defined as the unavailability of valid utterances for both participants. In our case this criterion should be extended because in a human-machine dialogue termination is up to the user (user-dependent). Therefore, we extend the conventional termination by a constraint that uses a special utterance which can be played by the `User`, called *the termination utterance*.

Definition 5.4.14 (Termination utterance). *An utterance u_n is the termination utterance if and only if* $\texttt{loc}(u_n) = \textsc{positive}$, $\texttt{id}(u_n) = n, \texttt{target}(u_n) = 1$, $\texttt{part}(u_n) = \texttt{U}$ *and* $\texttt{Content}(u_n) = \emptyset$.

Definition 5.4.15 (Termination). *Let d_n be a dialogue. d_n is a terminated dialogue if and only if u_n is the termination utterance.*

Since the main purpose of explanation dialogues is to get the `User` to understand a fact F the success of the dialogue is an important outcome. In other words, the dialogue system `EDS` should be able to determine whether the `Reasoner` could get the `User` to understand F or not.

Definition 5.4.16 (Success). *Let d_n be a terminated dialogue and let $F =$* $\texttt{Content}(u_1)$ *be the subject of d_n. If $F \notin \mathcal{US}$ then d_n is successful, otherwise it is not successful.*

A successful explanation dialogue is a dialogue where the `User` has understood the explanandum of the first explanation request (i.e. principle request). We can define a stronger criterion of success as follows:

Definition 5.4.17 (Strong success). *Let d_n be a terminated dialogue. If $\mathcal{US} = \emptyset$ then d_n is strongly successful.*

These criteria are interesting as they can be used in evaluating the efficiency of the `Reasoner`. Other criteria can be defined to precisely characterize the success. We limit ourselves to the aforementioned criteria.

Before closing the subsection, we find it relevant to highlight some possible problems about termination that arise if this dialogue is meant to be used between computational agents. We try to give general guidelines to avoid them.

Turntaking. The turntaking in our system is not deterministic (multiple-move). With computational agents we risk the occurrence of *starvation*, where one agent withholds the turn. To avoid such problem, one can make the turntaking explicit by specifying time slots or a fixed number of moves for each participant's turn. Other solution is to use the *threshold of interjection* where the agent gets the turn when she has a sufficiently strong desire to speak (Reed and Wells, 2007). This desire is computed by means of a function with respect to the history of the dialogue.

Freedom. Since participants are allowed to respond to early utterances in the dialogue we risk of *losing the point*. For instance, if the course of explanation was about why birds fly where the explainer explains aerodynamics then the explainee cannot ask about something which is not related to the previous explanation. A solution to this problem could be the application of a rigorous policy on replying. For instance, an agent is allowed to ask for another explanation if and only if it is related to a previous one.

Nested explanation requests. The explainee can request explanations about the content of other explanations. The risk relies in the possibility of getting into a loop of requesting and providing explanations. However this is not a problem when the explanatory model is finite as shown in (Arioua and Croitoru, 2015). If the explanatory model is infinite then constraints on the explanatory depth should be imposed.

5.4.5 The global picture

This subsection closes the section on the explanatory dialogue system `EDS`. We find it useful in this occasion to give a full picture on how an explanation dialogue works. We present the following three main stages where each of which describes how the dialogue is carried on:

- **Opening stage:** the `User` opens the dialogue by advancing the utterance $\textsc{explain}(\mathtt{U}, j, i, F)$ asking for an explanation about the fact F which is accessible and believed to be true by both parties.
- **Explanation stage:** in response to the explanation request in the opening stage, if the `Reasoner` cannot explain F then he utters $\textsc{inability}(\mathtt{R}, i, t, F)$ as a reply. If the `Reasoner` can explain then he issues $\textsc{attempt}(\mathtt{R}, i, t, E)$ that purports to explain the fact F. The utterance $\textsc{attempt}(\mathtt{R}, i, t, E)$ can be replied to by confirming or disacknowledging understanding. The former can be done by uttering

POSITIVE($\mathtt{U}, i, t, E$) such that E is the explanation provided in the first place. The latter can be done by stating NEGATIVE($\mathtt{U}, i, t, E$). Another interesting reply is EXPLAIN($\mathtt{U}, j, i, F'$) which is another explanation request about the provided explanation. This is similar to the case when one asks "Why the room is dark?" and has received the explanation "because there is no electricity", one can ask "why there is no electricity?". The two participants reserve the choice to reply to early utterances giving rise to new lines of discussions.

- **Closing stage:** the dialogue ends when the `User` advances the termination utterance or none of the participants cannot advance a valid utterance.

Following these three stages, we can perform an explanation dialogue about any subject taking into account the underlying background knowledge base. However, the background knowledge base may be inconsistent resulting in presenting competing or possibly implausible explanations. In the next section we extend `EDS` by argumentative faculties to facilitate explanations evaluation.

5.5 Argumentative Explanation Dialogues

The knowledge base of the `Reasoner` can be inconsistent due to different causes. Concerning our practical setting in DUR-DUR project, this knowledge base is collectively built by several knowledge engineers from different sites of the project. Due to various causes (errors in the factual information due to typos, erroneous databases/excel files, incomplete facts, unspoken obvious information "everybody knows" etc.) the collectively built knowledge base is prone to *inconsistencies*. This undermines the authority of the `Reasoner` as an explainer, therefore we can shift to argumentation whenever the explanation seems implausible or inconsistent with the `User`'s knowledge.

In this section we extend the `EDS` explanatory dialogue system with argumentative faculties. We aim at defining a minimal extension that includes argumentative locutions while keeping the dialogue system as simple as possible. We consider a subset of argumentative locutions from (Prakken, 2006), which are ARGUE and CONCEDE.

Definition 5.5.1 (Extended `EDS`). *Let $\mathcal{D}_{sys} = (\mathcal{P}r, \mathcal{C}, \mathfrak{R}, \mathcal{L}, \mathcal{K})$ be the explanatory dialogue system* `EDS`. *Recall that $\mathfrak{R}$ is the reply relation and $\mathcal{C} = \mathcal{C}_{\mathtt{R}} \cup \mathcal{C}_{\mathtt{U}}$ is the set of allowed locutions. We extend* `EDS` *as follows:*

- $\mathcal{C}_{\mathtt{R}} = \{\text{ATTEMPT}, \text{INABILITY}\} \cup \{\text{ARGUE}, \text{CONCEDE}\}$.
- $\mathcal{C}_{\mathtt{U}} = \{\text{EXPLAIN}, \text{POSITIVE}, \text{NEGATIVE}\}\ cup \{\text{ARGUE}, \text{CONCEDE}\}$.
- $\mathfrak{R} = \mathfrak{R} \cup \{(\text{ARGUE}, \text{ARGUE}), (\text{ARGUE}, \text{CONCEDE}), (\text{ATTEMPT}, \text{ARGUE})\}$.

Recall that $(\mathrm{X}, \mathrm{X}') \in \mathfrak{R}$ *means that the locution* X' *replies to* X.

These locutions define new utterances which are submitted to the same syntactical validity of utterances in Definition 5.4.3 in addition to the following.

Definition 5.5.2 (Utterance extension)**.**

- $u = \text{ARGUE}(x, i, t, a)$ *is an utterance if and only if the conditions 1-4 in Definition 5.4.3 are satisfied and a is an argument (cf. Definition 3.3.1, page 33).*
- $u = \text{CONCEDE}(x, i, t, A)$ *is an utterance if and only if the conditions 1-5 in Definition 5.4.3 are satisfied.*

As one can observe, the locution ARGUE replies to ARGUE. This is usual in argumentation dialogue where arguments counterattack other arguments. The CONCEDE locution expresses concession towards a claim advanced by an argument. The reply ARGUE to attempt is the *linking ring* between the argumentative locutions and the explanatory locutions. It allows the `User` to challenge the explanations advanced by the `Reasoner`. Such turn in an explanation dialogue can give rise to a pure argumentation dialogue where the other party can reply by ARGUE and so on and so forth until termination (concession). Nevertheless, this does not restrain the parties from replying to early utterances about other related explanations.

One may wonder about not adding the reply (ARGUE, EXPLAIN) to `EDS`, which asks for an explanation about the content of the advanced argument. In fact there is no theoretical reason that prevents us from doing so. Due to the dichotomy of syntax and semantic in `EDS` one can easily incorporate such extension. However, it will only violate the practical goal of providing a minimal extension of `EDS` that we set out in the beginning of the section.

Let us no proceed to the semantic part. As usual we need to give semantic conditions for utterances' content and to the way they reply to each other. For ARGUE(.) and CONCEDE(.) utterances it is straightforward as we shall see. However, the reply ARGUE(.) to ATTEMPT(.) need a formal characterization. It says in fact that an argument can *be in conflict* with an explanation. Since this particular relation between arguments and explanations has not been defined, let us introduce it.

Definition 5.5.3 (Conflict)**.** *Let $\mathcal{H} = (\mathcal{A}, \mathcal{X})$ be the corresponding argumentation framework of the background knowledge base $\mathcal{K}$. Let $a \in \mathcal{A}$ be an argument and E be an explanation built from $\mathcal{K}$ for a fact F. Then, we say a conflicts with E if and only if $C\ell^*_{\mathcal{R}}(\{\mathtt{Conc}(a), \mathtt{Facts}(E)\}) \models \bot$.*

An argument a is in conflict with an explanation E if the conclusion of the argument is inconsistent with the factual part of the explanation. It is to be noted that $E \notin \mathcal{A}$, i.e. E is not an argument and thus saying that $(a, E) \in \mathcal{X}$ is not correct. However, we may abuse notation and say a attacks E meaning that a conflicts with E.

Now let us introduce the new semantic conditions. Table 5.6 presents for the utterances ARGUE(.) and CONCEDE(.) the syntax and the informal meaning. The column "Effect" extends the effect rules defined in Definition 5.4.12 for each new utterance.

Note that these semantic conditions do not alter the previous ones, they only extend them. To sum up, the new EDS only extends the locutions, the reply relation and updates the semantic conditions that correspond to them. In Table 5.6:"Utterances", the utterances and their syntax are presented. Table 5.6:"Meaning" gives the meaning of the utterances. In Table 5.7:"Replies", the possible replies for each utterance are shown. Table 5.7:"Conditions" presents the semantic conditions for each reply and for each utterance.

Utterances	Meaning	Effects
ARGUE(x, i, t, a)	x responds by attacking an argument or an explanation	$\mathcal{CS}^i_x = \mathcal{CS}^{i-1}_x \cup \mathtt{Supp}(a) \cup \mathtt{Conc}(a)$, i.e. x becomes committed to the support and the conclusion of a.
CONCEDE(x, i, t, A)	concedes to the fact that A is true	$\mathcal{CS}^i_x = \mathcal{CS}^{i-1}_x \cup \{A\}$. x becomes committed to A.

Table 5.6: The new utterances and their meaning alongside to their effects on the stores. $x \in \{\mathtt{U}, \mathtt{R}\}$ and $\mathcal{CS}^i_x$ is the commitment store of x at stage i.

5.6 Dialogue Example

In this subsection we explain how the formal dialogue system EDS applies on Example 5.2.1 and 5.2.2. In Table 5.8 we follow step by step the advanced utterances and the evolution of the stores. The column i refers to

Utterances	Replies	Conditions
ATTEMPT$(\mathtt{R}, i, t, E)$	ARGUE$(\mathtt{U}, j, i, a)$	$a \in \mathcal{A}$ and a conflicts with E.
ARGUE(x, i, t, a)	ARGUE$(\overline{x}, j, i, b)$	$a, b \in \mathcal{A}$ and $(b, a) \in \mathcal{X}$.
ARGUE(x, i, t, a)	CONCEDE$(\overline{x}, j, i, A)$	$a \in \mathcal{A}$ and $A = \mathtt{Conc}(a)$ or $A \in \mathtt{Supp}(a)$.

Table 5.7: New replies and their semantic conditions. If $x \in \{\mathtt{U}, \mathtt{R}\}$ then $\overline{x} \in \{\mathtt{U}, \mathtt{R}\} \setminus \{x\}$. $j > i$ and $\mathcal{H} = (\mathcal{A}, \mathcal{X})$ is the corresponding argumentation framework of $\mathcal{K}$.

the identifier of the utterance, x refers to the participant, and while the column *Text* refers to the textual utterances, the column *Utterance* presents their formal counterpart. The column $\mathcal{CS}_{\mathtt{R}}$ denotes the commitment store of the `Reasoner`, $\mathcal{CS}_{\mathtt{U}}$ the commitment store of the `User` and $\mathcal{US}$ denotes the understanding store.

We do not give details on the logical formulas of each the utterance for space reasons. We give just an example to illustrate the idea. For instance:

$F_1 = \{perform(Stubbling, S_1)\}$

and:

$$
\begin{aligned}
E_1 = \{ & \\
& contains(S1, Stubble), \\
& contains(S1, Plant), \\
& residue(Plant), \\
& residue(Stubble), \\
& seedOn(S1), \\
& contains(S1, Stubble) \wedge contains(S1, Plant) \wedge residue(Plant) \wedge residue(Stubble)\} \wedge \\
& \quad seedOn(S1) \rightarrow perform(Stubbling, S_1) \\
& \}
\end{aligned}
$$

In the dialogue each utterance does not necessarily reply to the immediate precedent, it can reply to earlier utterances. For instance, the utterance NEGATIVE$(x, 6, 2, E_1)$ replies to the explanation at Stage (2).

The stores evolve due to utterances advancement following the effect rules of Definition 5.4.12. For instance, in Stage (1) we added F_1 to all the stores indicating that the two participants are committed to it and the `User` does not understand it. In Stage (8) we revoked F_1 from the understanding store because the `User` has understood the explanation at Stage (7) which explains F_1, the same thing happens for F_2 at Stage (5). This means that `U` now understands F_1 and F_2. At this stage if the dialogue ends then it is judged to be strongly successful because the understanding store is empty (cf. Definition 5.4.17 on strong success).

An alternative course of action is from Stage (7') to (10'), where we assumed that the `Reasoner` has advanced another explanation E_3 instead of E_2. The `User` advances an argument against the explanation E_3 to which the `Reasoner` has conceded. Then at Stage (10') the `User` declares that he is still incapable of understanding F_1 (because F_1 is still in the understanding store). At this stage, the `User` can leave and end the dialogue, or ask for other explanations.

After introducing the dialogue model of object-level dialectical explanations, in the next section we provide a case study of how they can be used in a real setting within the French national project DUR-DUR about Durum Wheat.

5.7 Use-case: Object-level Dialectical Explanations

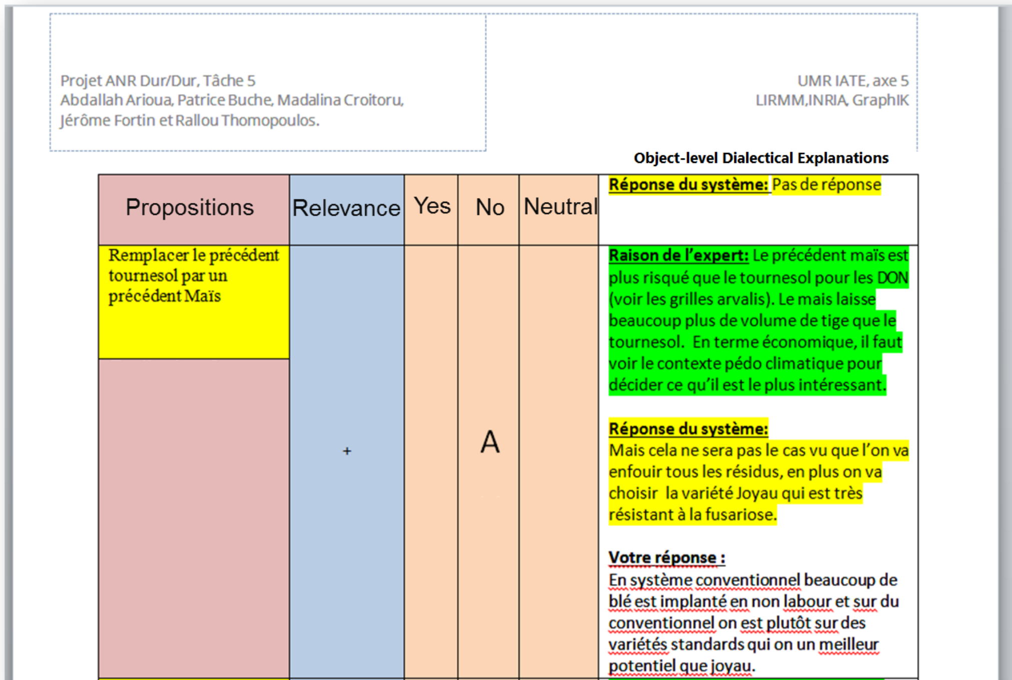
The DUR-DUR project suggests developing a systematic approach to investigate issues related to the management of the nitrogen, energy and contaminants, to guarantee a global quality of products throughout the production and the processing chain. One task in the project is to integrate multidisciplinary agronomy knowledge in a knowledge base. The Durum Wheat knowledge base has been constructed to fulfill such task.[8] It will be used in many computational tasks, notably analyzing and comparing the alternative innovative technical itineraries proposed in the project to reduce the use of chemical inputs (nitrogen fertilizers and pesticides). The content and architecture of the knowledge base will be more detailed in the next chapter. However some necessary information are needed for the use case.

The knowledge base is built by non-experts in Agronomy from reports and online materials which makes it prone to inconsistencies. Moreover, the construction was manual therefore the quality of data and the quantity were low. Inconsistencies occur as violations of constraints, for instance having the same scientific name for two different diseases. The goal of this use case is to show how object-level dialectical explanations can help in improving the quality of knowledge by reducing inconsistencies and improving the quantity by gaining new knowledge as a result of a dialogue with experts.

[8] Available online at `http://www.lirmm.fr/~arioua/dkb/`.

Table 5.8: The formalization of Example 5.2.1. Note that 7'-10' belong to Example 5.2.2 and the apostrophe is used to distinguish stages of different examples.

i	x	Text	Utterance	$\mathcal{CS}_{\mathtt{R}}$	$\mathcal{CS}_{\mathtt{U}}$	$\mathcal{US}$
1	U	Why do we perform stubble breaking?	$\text{EXPLAIN}(x, 1, 0, F_1)$	F_1	F_1	F_1
2	R	The fields have some stubble standing on the soil. Also, it they have residual plants which are left from previous crops. In order to be able to seed new crops we need to cut all stubble and remove residual plants.	$\text{ATTEMPT}(x, 2, 1, E_1)$	E_1	F_1	F_1
3	U	Why the fields have some stubble on the soil?	$\text{EXPLAIN}(x, 3, 2, F_2)$	E_1	F_1, F_2	F_1, F_2
4	R	After harvesting, some stubble stay incompletely cut.	$\text{ATTEMPT}(x, 4, 3, E_2)$	E_1	F_1, F_2	F_1, F_2
5	U	I understand.	$\text{POSITIVE}(x, 5, 4, E_2)$	E_1	F_1, F_2, E_1	F_1
6	U	But I still don't understand why do we perform stubble breaking.	$\text{NEGATIVE}(x, 6, 2, E_1)$	E_1	F_1, F_2, E_1	F_1
7	R	In fact without removing the stubble we cannot seed the next crop. Because the stubble will block the seeds from growing. Moreover, stubble breaking helps in creating a seedbed that will promote the germination of weed's seeds. Consequently, we will be able to remove these weed prior to the seeding of the new crop.	$\text{ATTEMPT}(x, 7, 1, E_2)$	E_1, E_2	F_1, F_2, E_1	F_1
8	R	Ok, I understand now why do we perform stubble breaking.	$\text{POSITIVE}(x, 8, 7, E_2)$	E_1, E_2	F_1, F_2, E_1, E_2	$\emptyset$
7'	U	In fact stubble breaking is used to fight against fungal diseases. That is why we perform it.	$\text{ATTEMPT}(x, 7', 1, E_3)$	E_1, E_3	F_1, F_2, E_1	F_1
8'	U	I do not think so because fungal diseases are caused by fungi which are microorganisms that are fought against by fungicides not buy mechanical machines.	$\text{ARGUE}(x, 8', 7', a)$	E_1, E_3	F_1, F_2, E_1, $\mathtt{Conc}(a), \mathtt{Supp}(a)$	F_1
9'	R	I concede.	$\text{CONCEDE}(x, 9', 8', \mathtt{Conc}(a) \cup \mathtt{Supp}(a))$	E_1, E_3, $\mathtt{Conc}(a), \mathtt{Supp}(a)$	F_1, F_2, E_1, $\mathtt{Conc}(a), \mathtt{Supp}(a)$	F_1
10'	U	I don't understand.	$\text{NEGATIVE}(x, 10', 7', E_1)$	E_1, E_3, $\mathtt{Conc}(a), \mathtt{Supp}(a)$	F_1, E_1, E_3, $\mathtt{Conc}(a), \mathtt{Supp}(a)$	F_1



Projet ANR Dur/Dur, Tâche 5
Abdallah Arioua, Patrice Buche, Madalina Croitoru,
Jérôme Fortin et Rallou Thomopoulos.

UMR IATE, axe 5
LIRMM,INRIA, GraphIK

Object-level Dialectical Explanations

Propositions	Relevance	Yes	No	Neutral	**Réponse du système:** Pas de réponse
Remplacer le précédent tournesol par un précédent Maïs	+		A		**Raison de l'expert:** Le précédent maïs est plus risqué que le tournesol pour les DON (voir les grilles arvalis). Le mais laisse beaucoup plus de volume de tige que le tournesol. En terme économique, il faut voir le contexte pédo climatique pour décider ce qu'il est le plus intéressant. **Réponse du système:** Mais cela ne sera pas le cas vu que l'on va enfouir tous les résidus, en plus on va choisir la variété Joyau qui est très résistant à la fusariose. **Votre réponse :** En système conventionnel beaucoup de blé est implanté en non labour et sur du conventionnel on est plutôt sur des variétés standards qui on un meilleur potentiel que joyau.

Figure 5.3: A screen shot of a small section of one of the forms (French).

We have carried out the use case with two Agronomy experts (of similar competences) within the project. While the number of experts is low in general for a given topic, it is even more difficult to perform use cases with domain experts of similar expertise in a project where everybody is chosen to complement the other. We have adapted an interaction protocol similar to the *Wizard of Oz* (Kelley, 1984), where the experts think they are interacting with an autonomous system, but in fact he/she is interacting with an unseen human being who simulates the intended behavior. This method is widely used in human-computer interaction, usability engineering, etc. So in general two forms (Figure 5.3) with identical structure have been prepared for the two experts which serve as a way to communicate with them. The communication has been conducted with the experts separately. We made sure that the experts do not report each others answers or discussions.

Each expert is presented with a set of 18 queries which are entailed by

Queries	Decision	Object-level Dialectical Explanations
Use a straw cereal precedent	No	1) *Expert*: *Since the aim is to reduce the use of chemical inputs, we are looking for a precedent that will allow us to fight against weed, provide nitrogen and reduce diseases. Quite the opposite of a cereal that will entertain a specialization of weed flora and leave little nitrogen in the soil and favor certain disease like septoria.* 2) `Reasoner`: *The introduction of such precedent limits the pressure of selection and generates a bigger range of flora with less density, thus facilitating weed control, hence less cost (financial + reduction of herbicides).* 3) *Expert*: *I do not really agree. In doing so it promotes a certain specialization of adventitious flora. This flora is going to be higher and higher and increasingly difficult to manage because it will have the same date of exercise of wheat.*

Table 5.9: An snippet of a partially filled form that shows a dialectical explanation between Expert 1 and the `Reasoner` along side to the decision. The expert here does not agree with the content of the query.

the knowledge base (universally accepted). Then they were asked to indicate whether they agree with the content of the query or not and provide an argument that supports their decision, they also had the choice to be neutral but still they were obliged to provide an argument. Moreover, they can ask other questions to inquire more, like "'Explain'. Then we took the experts' arguments for all 18 queries and generated counterarguments and explanations from the knowledge base, next we presented the counterarguments to the experts in a second round and asked them to either counterattack or concede or to acknowledge understanding. At the end of the communication with the experts we looked at the following:

- **Gain of new knowledge**: each argument advanced by the expert is analyzed to extract rules, fact and negative constraints. For instance, when the expert says (in the 1st line of the dialogue in Table 5.9) that straw cereal entertains a specialization of weed flora and leave little nitrogen in the soil and favor certain disease like septoria. This can be represented as a fact.

- **Reducing inconsistencies**: each argument advanced by the expert is analyzed to resolve inconsistency. To illustrate how inconsistencies are resolved, let us give an example. In the Durum Wheat knowledge

	Rules	Facts	Constraints	Inconsistencies
Expert 1	29	11	10	33
Expert 2	31	20	8	28

Table 5.10: The gain of new knowledge for Expert 1 and Expert 2.

base it is stated that: straw cereal precedent and sunflower precedent are used. The problem is that we cannot use the two precedents together. So this is a conflict in the knowledge base. In Table 5.9 the expert says that the aim is to reduce the use of chemical inputs (herbicide, fertilizers, etc.) and to fight against weed. These information when added to the knowledge base have allowed to conclude that the precedent in question is sunflower. Thus the conflict we had before has been solved, thus the straw cereal precedent is dismissed (delete the fact that stipulates the use of straw cereal).

Table 5.10 presents the number of rules, facts and negative constraints that have been elicited with Expert 1 & 2. It also presents (last column) the number of inconsistencies (i.e. number of minimal conflicts) in the KB. The KB had *49 inconsistencies* in total before carrying out of the use case.

The numbers in the first three columns refer to the new knowledge that has been elicited. The process of elicitation has been done manually by analyzing the forms. This new knowledge is fresh, that means it has not been found in the knowledge base. This is quite expected since the domain of Agronomy is vast and the proportion of what has been represented to what has not yet been represented is large. We can also observe the considerable quantity of rules we get compared to facts and negative constraints. This is explained by the fact that the experts had more tendency to give rules of thumbs instead of reporting data and facts.

In the *inconsistencies* column we observe that we could reduce the number of minimal conflicts by 16 with Expert 1 and by 21 with Expert 2. Note that the inconsistencies were independent, that means resolving one inconsistency does not result in resolving others inconsistencies. Therefore, what we are observing here is the direct effect of the experts' input on reducing inconsistencies.

In the same use case we asked the experts to give the decisions alongside with relevance values that represent to what extent the content of the query is relevant on a scale of:

	Revised decisions	Revised relevance values
Expert 1	25%	25%
Expert 2	25%	50%

Table 5.11: The percentage of revised decisions and revised relevance values.

{*very pertinent, pertinent, indifferent, not pertinent, not pertinent at all*}

We asked each expert to put the relevance after putting the decision and his response, then we asked them again to revise their relevance values and decisions after having received the response of the `Reasoner`. Then we looked how the values of relevance and decisions change.

The results is in Table 5.11 shows an average of 35% for Expert 1 and 25% for Expert 2. This indicates that about 7 and 5 propositions out of 18 have their decision and pertinence changed after the dialogue. The reason for this change does not mean that the expert was wrong and he/she was giving faulty information. But instead, the expert after receiving the response had more visibility on the content of the knowledge base through dialogue, hence he/she could better understand the content of the query, thus he/she revises his/her initial decision and relevance, note that the changes can be positive or negative. Put differently, it is possible that the expert agreed on the content of a query and he/she gives it a high relevance value. But after having the dialogue with the reasoner, the expert comes to realize that he/she should have not agreed on it.

Although being preliminary, this result shows that object-level dialectical explanations can help in better guiding the expert to understand the content of the knowledge base which would result in the improvement of the content of the knowledge base.

5.8 Conclusion

In this chapter we have presented the third contribution of the thesis. We have proposed a formal account of object-level dialectical explanations, which are explanations about the domain knowledge of a given knowledge base. We have proposed a dialectical account. That means a dialogue system for providing explanations called the `EDS` (explanatory dialogue system). This model instantiates Walton's `Explan` model and extends it along different directions. Table 5.12 shows the differences between our dialogue model and the `Explan` model. However, we should state clearly that we have not proposed a new system that is in rupture and in concurrence with Walton's model. It instead builds on it and extends it to cope with practical problems that one may encounter in logic-based settings.

Let us summarize the main characteristic of `EDS` model, others are mentioned in Table 5.12:

- **Fully formalized:** we followed the desiderata proposed in McBurney et al. (2002) and separated the protocol syntax and semantics. We have given a formal account of commitment and understanding stores and shown how they are manipulated.
- **Instantiated:** we have given a complete instantiation in a logical setting where the abstract concept of explanation is provided.
- **Multi-move and multi-reply:** the protocol of the dialogue is multi-move. That means that the participants can hold the turn freely until they hand it out to the other party. It is also multi-reply which means that an utterance can have multiple replies.
- **Liberal protocol:** alongside to being multi-move and multi-reply protocol, the participants can retrace and reply to early utterances.
- **Argumentative capabilities:** we can shift to argumentation whenever the explanation seems implausible or inconsistent with the `User`'s beliefs.

We have also discussed the issue of termination in Section 5.4.4 and we stated that `EDS` is used-dependent therefore the user is the one who determines termination. However, we provided some guidelines in case one would use `EDS` for agent-agent dialogues.

Finally, we have shown a use case of how object level-dialectical explanations can be used in knowledge acquisition and in reducing inconsistencies in the Durum Wheat Knowledge base.

In the next chapter we shift to the implementation of dialectical explanations, we explain the architecture of the DALEK system (DiALectical Explanation in Knowledge bases) that implements dialectical explanations in inconsistent knowledge bases. Next we present another contribution with respect to the DUR-DUR project where we provide a methodology to build Durum Wheat knowledge bases under the existential rules framework. We present the latest version of our Durum Wheat knowledge base that has been

Feature	`EDS`	`Explan`
Nested explanation requests	$\surd$	$\times$
Formal	$\surd$	$\times$ (Semi)
Questions asking	($\surd$)	$\surd$
Commitment stores	$\surd$	$\surd$
Understanding store	$\surd$	$\times$
Shift to Examination dialogue	$\times$	$\surd$
Argumentative capabilities	$\surd$	$\times$
Unique-move turntaking	$\times$	$\surd$
Multi-move turntaking	$\surd$	$\times$
Unique-reply	$\times$	$\surd$
Multi-reply	$\surd$	$\times$
Dichotomy of syntax and semantics	$\surd$	$\times$

Table 5.12: Comparison between `EDS` and `Explan`. $\surd$ means that the feature is available in the dialogue model, $\times$ means that it is unavailable. ($\surd$) means it is available within the logical setting but not within the dialogue model.

improved by the object-level dialectical explanations studied in the use case presented in this chapter.

6 Application

In this chapter we present two outcomes of the thesis. The first one is related to the project DUR-DUR where we present a methodology of constructing Durum Wheat knowledge bases in the expressive framework of existential rules. The methodology emphasizes on how we conceptualize the knowledge base and how to represent knowledge using conceptual graphs. Then we show how to test whether the knowledge represented in the knowledge base will allow query answering by checking whether the rule base provokes undecidability or not. We explain this methodology on the Durum Wheat knowledge base which has been built for the project. The second outcome is the DALEK prototype that implements dialectical explanations to explain query answers in inconsistent knowledge bases. We present its architecture in details.

6.1 Dur-Dur and the Durum Wheat Knowledge Base

The Durum Wheat knowledge base has been constructed within the French National Project DUR-DUR. The goal of this knowledge base is to integrate scientific knowledge acquired from different tasks during the project to redesign the durum wheat chain. The DUR-DUR project suggests developing a systematic approach to investigate issues related to the management of the nitrogen, energy and contaminants, to guarantee a global quality of products throughout the production and the processing chain. Started in 2014 and planned over 4 years, the project aims at integrating the 3 dimensions of the sustainability (environmental, economic, and social), at 4 levels of investigation (4 tasks) with a complementary task (task 5). Figure 6.1 depicts the different tasks of the project where the fifth task's central role is to integrate knowledge from different tasks. The Durum Wheat knowledge base is on of the products of the fifth Task. It will be used in many computational tasks, notably analyzing and comparing the alternative innovative

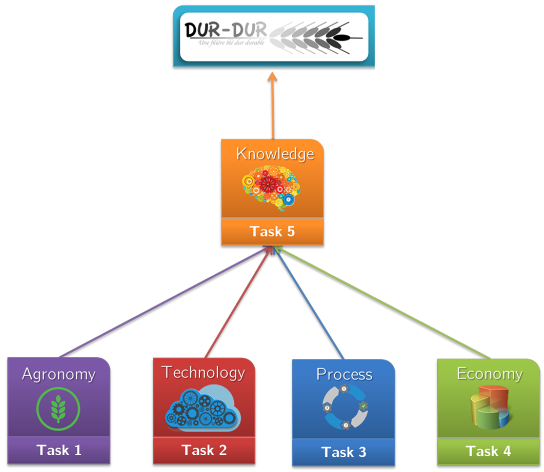


Figure 6.1: The different tasks of the Dur-Dur project. The knowledge tasks aims at integrating multidisciplinary knowledge from other tasks.

technical itineraries proposed in the project to reduce the use of chemical inputs (nitrogen fertilizers and pesticides). The knowledge base represents domain-specific knowledge about Agronomy. It is composed of four main parts:

- **Vocabulary:** it contains knowledge about concepts and relations.
- **Rules:** they represent rules that encode generic knowledge.
- **Negative constraints:** this part contains constraints about crops and Agronomy-related constraints.
- **Facts:** this part contains factual knowledge about Agronomy-related subjects (fertilizers, pesticides, diseases, etc.).

In the next section we start by highlighting the guidelines which were followed to author the knowledge base (Subsection 6.1.1) then we turn to the the internal structure of the knowledge base including the *vocabulary*, the *rule-base* alongside with the *constraints* and the *factual knowledge* (Subsection 6.1.2).

6.1.1 The authoring

A multidisciplinary process of knowledge acquisition and representation was deployed to author the knowledge base. We used technical reports to define

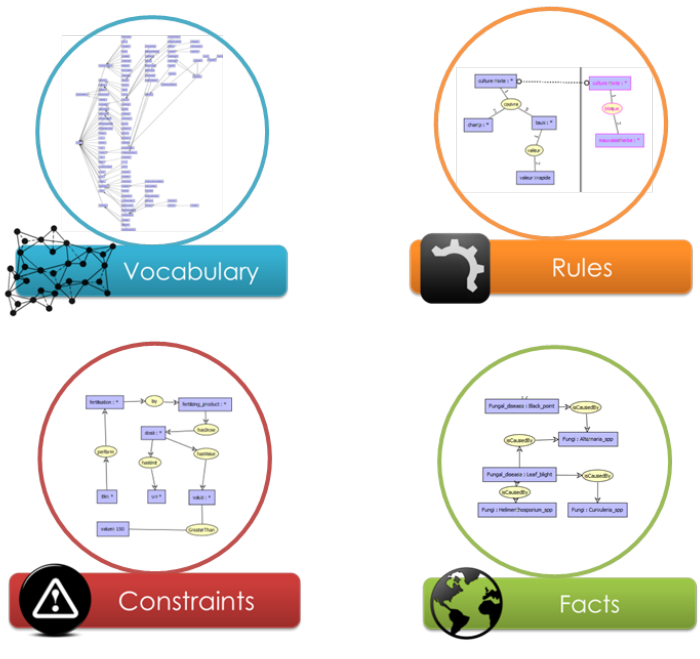


Figure 6.2: An overview of the Durum Wheat knowledge base. The circles contain knowledge examples represented in the conceptual graph framework.

the scope of the knowledge base and the relevant concepts of our vocabulary. Taking into account the recommendation of Thunkijjanukij et al. (2008). We followed three steps *specification, conceptualization and formalization* to build the knowledge base.

Specification. The scope of the Durum Wheat knowledge base has been defined by exclusively focusing on *Durum Wheat Sustainability* management. The goal is improving Durum Wheat sustainability in France and reduce the use of nitrogen fertilizers and pesticides and optimize energy consumption using a systematic approach that makes use of innovative technical itineraries. The contribution of the knowledge base lays in offering an expressive way of representing domain-knowledge.

Conceptualization. The concepts and the relations among them alongside to rules, facts and constraints have been defined and collected from technical reports (see Figures 6.3 and 6.4) and online materials[1] It is worth mentioning that in the vocabulary part we have built on the vocabulary of Agropedia indica (Sini and Yadav, 2009) with an increase (and modification) in content that approximates 60%.[2]

Formalization. Since understanding logical formulas is quite difficult for experts who are not familiar with KRR formalism we have chosen a graphical framework (Conceptual Graphs (Sowa, 1976, 1983)) to author the knowledge base. Moreover, the conceptual graphs (CGs) made it easy for the Agronomy experts to understand the content of the knowledge base. Furthermore, CGs enjoy the same expressive power as existential rules. In fact, it is an equivalent formalism of existential rules as shown in Chein and Mugnier (2009). Therefore, our choice was to choose CGs for knowledge acquisition and the existential rules as a framework for theoretical study. For CGs, we used *CoGui 1.6b* which is an IDE for representing and reasoning with CGs.[3] We shall explain in-depth in Section 6.1.2 the graphical and logical representation for each part of the knowledge base. The facts within the knowledge base are exported to an RDF/XML format whereas the vocabulary, rules and constraints are exported as DLGP format (**D**ata**L**o**G** **P**lus (Leclère et al., 2013)). The vocabulary of the knowledge base contains 279 concepts and

[1] see `http://www.fiches.arvalis-infos.fr/`.
[2] `http://www.agropedia.net`.
[3] `http://www.lirmm.fr/cogui/`, GraphIK, LIRMM.

Variété	Obtenteur	Type de gluténines*	Rdt (% du témoin)	Maladies feuilles	Fusa.	Besoin N (kgN/q)
Pescadou	FD	G3	103%	-	+	3.5
	Remarque : Absorbe bien l'N en fin de cycle					
Dakter	LG	G3	90%	+	-	3.5
	Remarque : Potentiel plus faible mais suffisant pour atteindre une production de 50 à 60 q/ha					
Miradoux	FD	G3	100%	-	-- (DON)	3.7
	Remarque : Variété bien connue. A noter que Miradoux est la principale variété cultivée dans les différentes régions et pourra donc servir pour notre témoin des ITKs de référence régionaux et du N0 et du Phyto++ ce qui permettra de fournir des lots à l'UMR IATE avec différents niveaux de teneur en protéines					
Isildur	R2N	G1	95%	+	-	?
Fabulis	LG	G4	98%	-	+	?
Anvergur	RAGT	?	113%	++	-	3.7
	Remarque : Potentiellement la prochaine référence car combine différentes caractéristiques.					
Joyau	SYN	G2	90%	-	++	3.7
Auris	LG	G3	?	+	-	?
	Remarque : Peu de surfaces cultivées					

Figure 6.3: Some structured data about Durum Wheat varieties.

1.1. ITK « Référence »

- Variété Miradoux
- Précédent tournesol
- Déchaumage
- Semis à 280 grains /m² autour du 20-25 octobre
- Fertilisation : 40U au tallage, 50U fin tallage (avant E1cm), 50U à 1 nœud et 60U DFL ou gonflement
- Désherbage : A l'automne si risque fort ou si problème de résistance type raygrass en privilégiant dans ce cas des herbicides racinaires aux herbicides foliaires. Complément de désherbage en sortie hiver si besoin avec un herbicide foliaire en fonction des types de résistance rencontrés
- Fongicides : 3 traitements (sous règles de décision) à savoir un premier à 2 nœuds, 1 second à Dernière Feuille Etalée contre la rouille et la septoriose) et un à la Floraison contre la fusariose
- Insecticide si besoin sous règle de décision + molluscicide si besoin à 3 feuilles/tallage

Figure 6.4: Description of how to cultivate Durum Wheat.

116 relations, the rule-base contains 23 rules and the constraints part contains 25 constraints. The factual part has around 900 atoms. The knowledge base is available online at `http://www.lirmm.fr/~arioua/dkb/` where the reader can find downloadable materials.

6.1.2 The structure

As depicted in Figure 6.2 the knowledge base is composed of four parts. It is worth mentioning that on the logical level the vocabulary and the rule-base are the same. However, we adapt here the Semantic web notation and we differentiate between them. Therefore, we distinguish between those rules that express logical consequences (in the rule-base) and those that encode generalizations and classes inclusions (in the vocabulary).

6.1.2.1 The vocabulary

The vocabulary represents an explicit specification of the terms and concepts used in Agronomy. The vocabulary is composed of two parts: (1) concept types hierarchy and (2) relation types hierarchy.

1. **Concept types hierarchy:** concepts are organized within a hierarchy as super-concepts and sub-concepts. For instance, the concept disease and its sub-concepts (e.g. viral disease, fungal disease, etc.), types of pesticides (e.g. herbicide, insecticide, fungicide) are all of organized in a hierarchy.

2. **Relation types hierarchy:** in CGs the concepts are related by relationships. Since concepts are divided into super-concepts and sub-concepts, relationships are divided in the same way. In the relation types hierarchy we find super-relations and sub-relations. For instance, the relation "useSowingProcess" which relates the *seeding and sowing* production step with the process of *sowing* (which is a super-concept of *broadcasting*, *behind plough* and a sub-concept of *process*). This relation is a sub-relation of the super-relation "useProcess" that relates any production step with any process.

In CGs the hierarchy of concept types is represented as in the upper graph of Figure 6.5. Rectangles represent concepts and the arrow represents the generalization between them where the source of the arrow is the sub-concept and the target of the arrow is super-concept. In the relation types hierarchy (the lower graph), the circles are the relations and the arrows are generalizations.

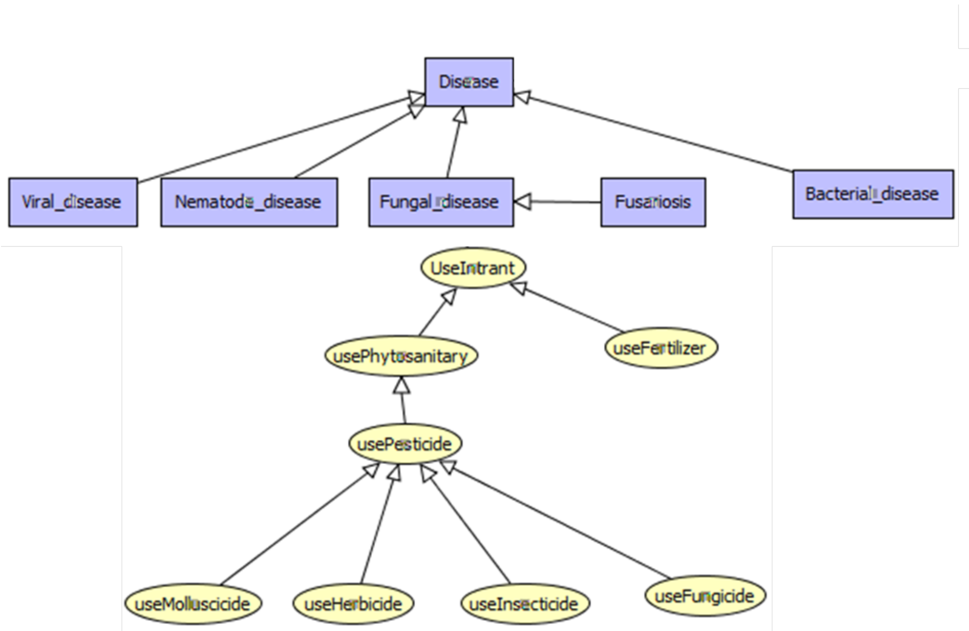


Figure 6.5: Concept and relation types hierarchy.

To better illustrate the relation between existential rules and CGs, let us take an example that shows the transformation of some part of the graphs of Figure 6.5 to their logical form.

Example 6.1.1. *The left-most part of the concept types hierarchy that indicates that "Viral disease is a disease" is represented logically by a rule as follows:*

- $\forall x(Viral\_disease(x) \rightarrow Disease(x))$.

The part of the relation types hierarchy that indicates that "Using Herbicide is using Pesticide" is represented logically by a rule as follows:

- $\forall x \forall y(useHerbicide(x, y) \rightarrow usePesticide(x, y))$.

6.1.2.2 The factual knowledge

In the Durum Wheat knowledge base the factual part represents domain-specific knowledge. This knowledge is divided into two parts: (1) general factual knowledge and (2) knowledge about different technical itineraries.

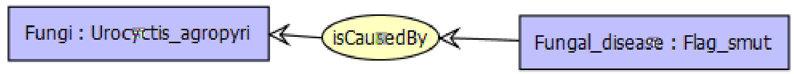


Figure 6.6: The set of facts (d) about fungal diseases and fungus.

According to Sebillotte (1978) a *technical itinerary* is a "*logical organized course of technical actions* applied to a cropped species".

General factual knowledge is the part of the knowledge base that represents general facts about the domain, for instance, *Miradoux* is a variety of Durum Wheat or the fungal disease *Fusarium Flag smut* is caused by, among other causes, the fungi *Urocyctis agropyri* of the family Fusarium. The following is an example of a set of facts. Recall that commas are interpreted as conjunctions.

(d) $\{Fungal\_disease(Flag\_smut), isCausedBy(Fusarium\_ear\_blight, Urocyctis\_agropyri), fungi(Urocyctis\_agropyri)\}$.

Here we have the relation *isCausedBy* instantiated on the individuals $Flag\_smut$ and $Urocyctis\_agropyri$. The former is a fungal disease as stated by the concept $Fungal\_disease$ and the latter is a fungi. Figure 6.6 depicts the set of facts in the conceptual graph framework. In conceptual graphs the rectangles are called *concept nodes* and the circles are called *relation nodes*. A concept node has a *concept type* and a *marker* which can be either an *individual marker* (constant) or a *generic marker* (a variable denoted as *).

The second part of the factual knowledge part are those facts about the technical itineraries. In what follows we give a real-world example of a well-known technical itinerary in France.

Example 6.1.2. *This example represents the reference technical itinerary in France which is followed by farmers to cultivate their fields.*

"The variety to be seeded in the soil is Miradoux, the culture precedent is sunflower. The soil is prepared by means of harrowing. The seeding is done with density of 280 grains/m^2. Fertilization is to be performed at the growing stage when the tiller begins with dose 40u and 50u at the end of the tiller."

This technical itinerary is a set of facts, e.g. "variety is Miradoux", "Fertilization is to be performed at the growing stage", etc. However, not any set of facts. Particularly, it is a precise set of describing facts. Actually, any ITK (according to the studied reports) should precisely account for the following steps:

1. Variety to be seeded.
2. Date of seeding alongside the density.
3. Cultural precedent.
4. Inter-cropping techniques.
5. Soil preparation method.
6. Disease management method.
7. Weed management method.
8. Insect control method.

Thus a technical itinerary should be mainly composed of these describing facts. The following is a snippet of the technical itinerary described in Example 6.1.2.

$$\mathcal{F}_{ITK} = \begin{cases} Soil(Soil_1) & Durum\_wheat(D1) \\ isOfVariety(D1, Miradoux) & Variety(Miradoux) \\ isCultivatedOn(D1, Soil_1) & Seeding\_and\_sowing(Seeding_1) \\ Seed(Seed_1) & useSeed(Seeding1, Seed_1) \\ seedOf(Seed_1, Durum_{w1}) & isAppliedOn(Seeding_1, Soil_1) \\ withDensity(Seeding_1, Density_1) & Density(Density_1) \\ Unit(grain\_mm) & hasValue(280) \\ Value(280) & \end{cases}$$

6.1.2.3 The rule-base

Rules in the rule-base encode general-purpose domain-specific knowledge. For instance, consider the following rules:

Example 6.1.3 (Example of rules)**.**

(a) *If a Durum Wheat x has fusariosis disease y then there exists a mycotoxin z that has contaminated the Durum Wheat x.*

$\forall x, y(Durum\_wheat(x) \wedge hasDisease(x, y) \wedge Fusariosis(y) \rightarrow$
$\exists z\ isContaminatedBy(x, z) \wedge Mycotoxin(z))$

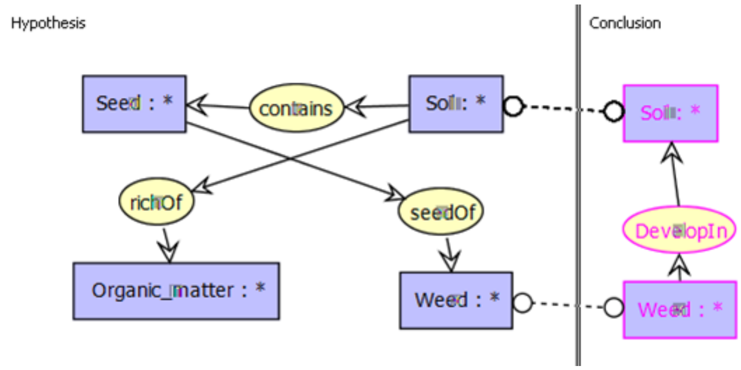


Figure 6.7: The rule (b) in the CGs framework.

(b) *If the soil is rich of organic matters and it contains seeds of weed then these seeds will develop in this soil.*

$\forall x, y, z, w(Soil(x) \wedge Organic\_matter(y) \wedge richOf(x, y) \wedge contains(x, z) \wedge Seed(z) \wedge seedOf(z, w) \wedge Weed(w) \rightarrow developIn(w, x))$

The mycotoxin z is unknown (it could be Aflatoxins, Deoxynivalenol, etc.) but still the information that "there is necessarily a mycotoxin" is present, which is an important information when it comes to risk management where a possible contamination by any mycotoxin is taken to be critical. Moreover, the importance of such representation manifests also in helping knowledge elicitation where the knowledge base can make use of incomplete information and then be updated incrementally by identifying the existential variables.

In conceptual graphs, rules (e.g. rule (b)) are composed of a hypothesis (left) and a conclusion (right) which are conceptual graphs of facts with generic markers (*). The dashed lines link those concepts that share the same variables (called frontier variables). That means, variables that appear in the hypothesis and in the conclusion. In the rule (b), the concept *weed* in the hypothesis part shares the same variable with the concept *weed* in the conclusion.

As it is explained in Chapter 3 in the existential rules framework, certain classes of existential rules render the inference undecidable. However, there are some classes that ensure decidability. Most notably, FUS (Finite Unification Set) and FES (Finite Expansion Set) classes.[4] The online tool *Kiabora* deploys syntactical and semantic analysis on any set of rules writ-

[4]For more information about the classes see Baget et al. (2011b).

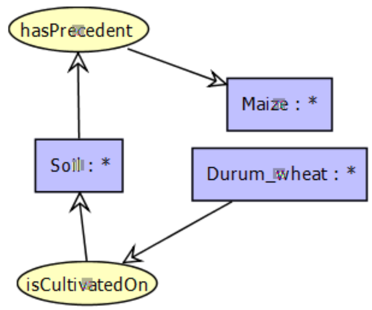


Figure 6.8: The negative constraint (c) in the CGs framework.

ten in the DLGP format.[5] The tool, for a given rule-base, classifies all the rules with respect to the known classes. From the analysis we found that our rule-base lays within the decidable classes. Specifically, FUS and FES.

6.1.2.4 The negative constraints

Representing what cannot be allowed within certain domain of interest is called *negative constraint* (or constraint). Consider the following negative constraint:

Example 6.1.4 (Negative constraint).

(c) $\forall x, y, z(Soil(x) \wedge Maize(y) \wedge Durum\_wheat(z) \wedge hasPrecedent(x, y) \wedge isCultivatedOn(z, x) \rightarrow \perp)$.

This negative constraint forbids using Maize as a precedent on a soil if we want to cultivate Durum Wheat on this soil. Figure 6.8 represents the CGs representation of this negative constraint.

Besides this type of constraints we have the banned types constraints. These are particular forms of constraints that express concept disjointness. For instance, a soil x cannot be a disease, $\forall x(Soil(x) \wedge disease(x) \rightarrow \perp)$. In the Durum Wheat knowledge base, all concepts are disjoint except those concepts which have a generalization/specialization relations among them.

[5] Kiabora 0.1 website: `http://www.lirmm.fr/~mugnier/graphik/kiabora/`, see Leclère et al. (2013) for a detailed explanation.

6.2 The DALEK Prototype: Explain!

In this section we present the general guidelines that we followed to implement dialectical explanations. The DALEK prototype implements meta-level as well as object-level dialectical explanations. DALEK is programmed in a way that it captures a standalone argumentation dialogue. It engages a `User` and the `Reasoner` in a dialogue about the entailment of any boolean conjunctive query in inconsistent knowledge bases represented within the language of existential rules. Moreover, DALEK implements commitments and understanding stores.

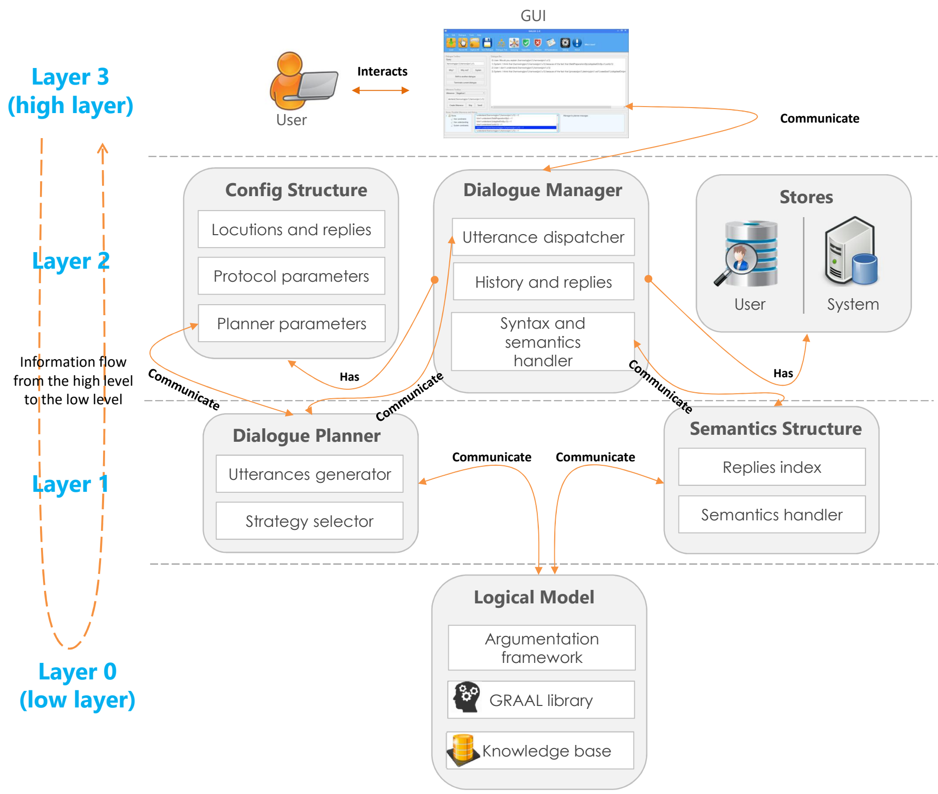


Figure 6.9: Layered architecture of DALEK.

Figure 6.9 presents the layered architecture of DALEK. Each layer is composed of modules and each module is composed of sub-modules. The information flow passes from the layer 3 to layer 0 through the intermedi-

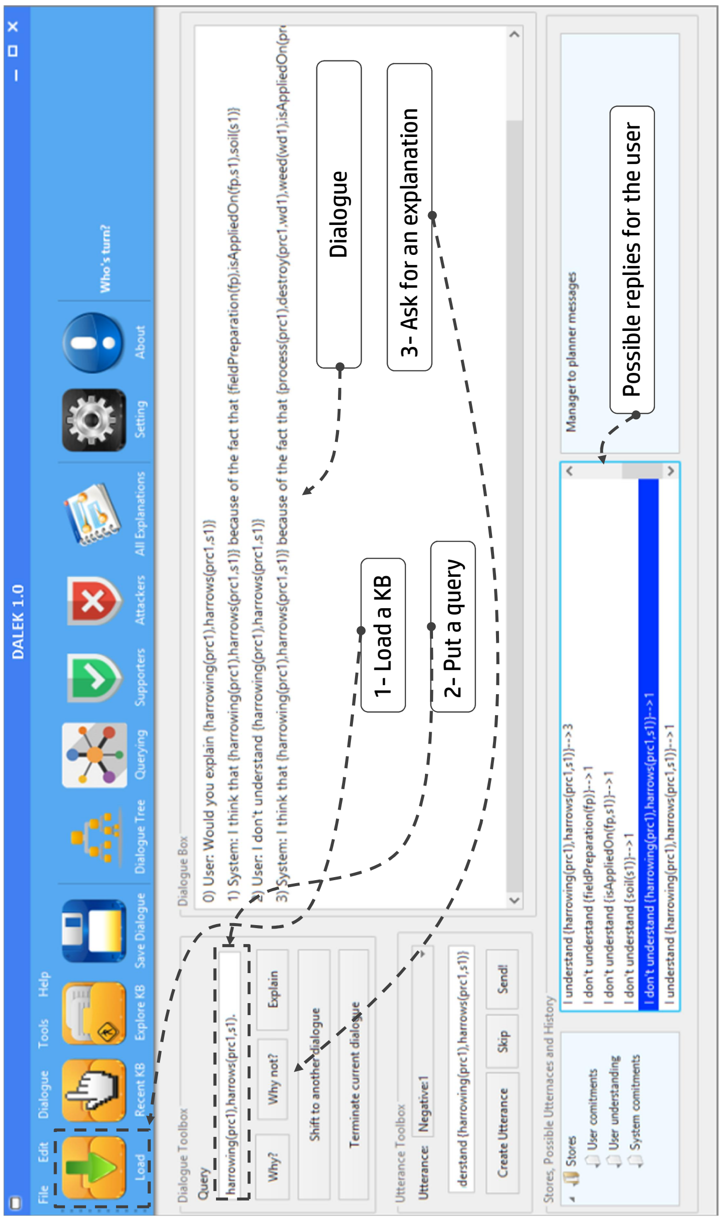


Figure 6.10: The GUI of DALEK while carrying out a dialogue with a user.

ate layers using the "communicate" link between modules. The "has" link symbolizes possession. Note that sub-modules intercommunicate by default.

- Layer 3 (high layer): the graphical user interface.
- Layer 2: dialogue manager, configuration structure and stores.
- Layer 1: dialogue planner and semantics structure.
- Layer 0 (low layer): logical model.

As depicted in Figure 6.9 when the `User` interacts with the GUI (Figure 6.10), the latter communicates with *the dialogue manager* which possesses the *configuration structure* and the *stores*. Then, the dialogue manager, at its turn, communicates with *the semantics structure* through the sub-module "Syntax and semantics handler" and with *the dialogue planner* through the sub-module "Utterance dispatcher". Next, *the dialogue planner* and *the semantics structure* communicate directly with *the logical model* that uses the *Datalog*± GRAAL library Baget et al. (2015) to query *the knowledge base*. In what follows we detail each module.

6.2.1 Graphical user interface

The graphical user interface (Figure 6.10) allows the user to load a knowledge base written in DLGP format. It allows also to load the most recent knowledge base. In the first case DALEK compiles the knowledge base into another structure that indexes the facts and their derivation paths. In the second case, the structure is only loaded as it was already computed before. Then we can put the query directly in the query box and ask for an explanation. To know whether the query has an answer or not one can click on *Querying* button and select the answer variables of the query and get the results. The buttons *Supporters*, *Attackers* and *All Explanations* give the set of supporters, attackers of the supporters and explanations of the query (in the sense of Chapter 5, Definition 5.4.8).

After putting the query in the query box one can start a pure argumentation dialogue using the button *Why* and *Why not* or a pure explanation dialogue by clicking on *Explain*. To mix the two dialogues, one has to click on *Shift to another dialogue*. After choosing the type of the dialogue, the user can reply to the `Reasoner` responses either by choosing already computed response or creating a new response through the *Utterance Toolbox*. While doing the dialogue the user can visualize the history of the dialogue by clicking on *Dialogue Tree* where the dialogue will be shown in form of a tree.

6.2.2 Configuration structure

This module is responsible for holding the information about the different parameters of the dialogue. It specifies: (1) the set of allowed locutions (e.g. ATTEMPT, POSITIVE, etc.) alongside with their legal replies, (2) the parameters of the protocol, e.g. unique-move, multiple-move, unique-reply, multiple-reply, the participants, etc. and (3) the parameters of the planner, e.g. types of strategies, utterance selection criteria, etc. To facilitate interaction with the `User`, the current version of DALEK adapts a *unique-move and unique-reply protocol*. These settings can be changed in the configuration structure with additional minor modifications in the planner.

6.2.3 Stores

A commitment store is a set of formulas to which a participant is committed. An understanding store is a set of formulas which a participant has not yet understood. The stores are modified by certain utterances. This module is responsible to manage these stores. They are kept in memory as small knowledge bases. The store can also directly be consulted and altered by the `User`.

6.2.4 Dialogue manager

The dialogue manager is the referee between the `User` and the `Reasoner` (i.e. dialogue planner), it dispatches their utterances through the sub-module "Utterance dispatcher" after ensuring their legality. To verify the legality, the dialogue manager communicates with the module semantics structure through the sub-module "Syntax and semantics handler" that makes use of the stores. Here is a brief description of the verification steps.[6]

- **Syntactical verification:** It ensures the legality of any advanced utterance with respect to : (1) legality of the utterance itself, and (2) legality of the reply within the dialogue. The first one checks whether it is the turn of the speaker or not and whether the id of the utterance is correct. It also checks whether the used locution is correct. The second one checks whether the utterance is a correct reply to the previous one by checking membership in the "Locutions and replies" of the configuration structure module.

[6] Each description is not necessarily exhaustive. However, it follows the protocol of dialectical explanations shown in previous chapters.

- **Semantics verification:** It ensures the legality of the utterances with respect to the content. It checks whether the advanced utterance holds a semantically valid content (e.g. EXPLAIN should hold a why question, ATTEMPT should hold an explanation, etc.) and it replies with a semantically valid content. This procedure is ensured by the semantics structure. A final verification is to check the understanding store of the `User` to avoid circular explanations after playing the utterance (this is ensured by "Syntax and semantics handler"). If the configuration of the dialogue imposes consistency of commitment stores then the semantics verification checks whether the commitment stores of the participants will maintain consistency after playing the utterance.

Since the dialogue is asymmetric, that means the `User` does not have the possibility to interact with the `Reasoner` using natural language. The dialogue manager proposes to the user the possible reply that he/she can use to respond to the `Reasoner`.

6.2.5 Semantics structure

This structure implements an *operational semantics* of the dialogue. It associates with each reply a procedure that should be called by the dialogue manager to check the legality of the reply. For instance, when presented with an utterance ARGUE$(\mathtt{U}, 7, 4, b)$ that responds to ARGUE$(\mathtt{R}, 4, 3, a)$, the semantics structure first gets the corresponding procedure (i.e. *argueToargue* reply procedure) then checks whether B is an argument, next it verifies whether B attacks A by communicating with the logical model.

6.2.6 Dialogue planner

This module represents the `Reasoner`. It receives the utterances from the `User` through the dialogue manager and plans the next utterance to advance. The planner in its current state follows a simple profile, a *follow-through* strategy where it tries to answer `User`'s utterances as they come. The planner also performs the following tasks (among others):

- **Explanation computation:** When an explanation is requested for a query $\mathcal{Q}$, the planer asks the logical model to retrieve the rules and set of facts that can deduce the query in a backward-chaining manner. Using such technique we can insure consistency and minimality of explanations.

- **Argument computation:** When there is the possibility to attack an explanation or counterattack an argument, the `Reasoner` asks the logical module to compute all possible arguments then it chooses one of them according to a specific strategy. In our case we just use a random selection function.

6.2.7 Logical model

Since the DALEK prototype uses the existential rules language to represent knowledge bases, we have used the GRAAL library (Baget et al., 2015) as a underlying engine for reasoning. The sub-module "Argumentation framework" uses GRAAL and computes argument, counterarguments in the fly using query rewriting techniques. It is to be noted that the logical model faces the problem of intractability when we scale up to bigger knowledge bases.

- **Minimal conflicts computation:** This procedure takes the set of negative constraints and find for each negative constraint the set of facts that triggers it (i.e. a minimal conflict, c.f page 3.2.14, Definition 3.2.14). If there is such set, it stores it in a file called *conflicts*. This file is in fact a physical representation of a hypergraph called the conflict hypergraph (Chomicki et al., 2004).

- **Argument generation:** This task computes an argument given a query $\mathcal{Q}$. It computes by backward-chaining (using the rewriting algorithm of (König et al., 2015)) the set of facts involved in the deduction of $\mathcal{Q}$. This derivation forms an argument for $\mathcal{Q}$.

- **Counterargument generation:** This task receives an argument a and computes its counterarguments. It proceeds by getting all elements of its hypothesis, then it looks up in the conflict hypergraph for those subsets of the hypothesis that are involved in any conflict. If so, the procedure constructs a counterargument from the conflict's graph, otherwise it returns an empty set.

In the next section we show how DALEK is used in knowledge acquisition within the domain of Durum Wheat sustainability improvement.

6.3 Conclusion

In Section 6.1 we have shown another contribution regarding to the project DUR-DUR which consists of the a methodology of constructing domain-specific knowledge base about Durum Wheat. To show the implementability of dialectical explanations and its use in real-world setting, we presented in Section 6.2 the general guidelines of a prototype called DALEK that implements dialectical explanations.

7
Conclusion and Perspectives

This chapter concludes the thesis and presents several possible directions for future work.

7.1 Conclusion

In the beginning of the thesis we set out the following research problem.

> Research problem
> *How do we make Consistent Query Answering intelligible to the end-user?*

Our thesis is that:

> *Formal Argumentation can serve as a solution to this problem. Moreover, it provides an added explanatory value with respect to the state-of-the-art approaches.*

To validate the thesis we provided three main contributions within the framework of logic-based argumentation under existential rules.

- One-shot Argument-based Explanations (Chapter 3).
- Meta-level Dialectical Explanations (Chapter 4).
- Object-level Dialectical Explanations (Chapter 5).

In the first contribution we first provided a study of logic-based argumentation under existential rules which revealed interesting properties such as finiteness, coherence, relative groundedness and non-triviality. We also characterized the outputs of this argumentation framework by relating universal acceptance to the concepts of blocks and proponent sets. A block is an admissible set of arguments that attacks all the supporters. A proponent set is a set of supporting arguments that are distributed over all the extensions. We related the problem of finding a block or a proponent set to the problem of finding hitting sets. We established a representation theorem between the existence of a block (resp. proponent set) and the non-universal (resp. universal) acceptance of a query. Since blocks and proponent sets are necessary and sufficient reasons to determine (non-)universal acceptance we considered them as *One-shot Argument-based Explanations*.

In the second contribution we have shown that logic-based argumentation can go further than other approaches. Indeed, by exploiting the equivalence between CQA semantics and universal acceptance, logic-based argumentation provides elaborated and engaging explanations called *Meta-level Dialectical Explanations*. This type of explanations is based on a dialectical proof theory that we proposed for universal acceptance. This makes the contribution twofold:

- We proposed a new dialectical proof theory for logic-based argumentation.
- We provided an explanation mechanism that overcomes the issues raised about One-shot Argument-based explanations without the need to change the framework, thanks to formal dialectics.

To investigate the real effect of such improvement we have conducted an experimental evaluation on the impact of meta-level dialectical explanations compared to one-shot argument-based explanations on users regarding the following aspects:

- **Accuracy:** we looked whether meta-level dialectical explanations would make the user better understand inconsistent situations. We made the hypothesis that his/her accuracy would increase when asked new queries about the inconsistent situation.
- **Time:** we looked whether meta-level dialectical explanations would make the user faster in answering new queries.
- **Appreciation:** we looked what is better appreciated, meta-level dialectal explanations or one-shot argument-based explanations on a scale of “not clear at all”, “not clear”, “clear”, “so-so”, “very clear”.

On the accuracy aspect, the data have confirmed our hypothesis as we found that those who received meta-level dialectical explanations were more accurate than those who received one-shot argument-based explanations. On the time aspect, results have shown no effects. It is to be noted that the absence of effects is more likely to be the result of insufficient number of subjects. On the appreciation aspect, we have found that meta-level dialectal explanations are better appreciated than one-shot argument-based explanations where the median of the former is “clear” and the one of the latter is “so-so”.

The post-hoc analysis has shown interesting pattern of long and well-structured justifications when the users[1] are asked to justify their answers. This post-hoc analysis can be further investigated in the future.

In the third contribution we provided an abstract dialogue model of explanation. We have shown how it improves Walton’s model (Walton, 2011) on different aspects. The instantiation of this model on the existential rules framework yielded Object-level Dialectical Explanations. A use-case with Agronomy experts has revealed interesting preliminary results on the utility of this type of explanation regarding knowledge acquisition and inconsistency resolution. The subject of the use-case was the Durum Wheat knowledge base which we have built within the DUR-DUR project.

To show the feasibility of these explanations, we have described the architecture of a prototype called DALEK that implements One-shot Argument-based Explanation, Object-level Dialectical Explanations and certain aspects of Meta-level Dialectical Explanations.

7.2 Future work

In this section we present some directions for future work for our contributions.

Logic-based Argumentation in Existential Rules. In Chapter 3 we have presented a logic-based instantiation of Dung’s abstract model in the existential rules framework. Although interesting properties have been studied some issues are still to be investigated and handled. We present them hereafter in a prioritized order:

- *Algorithmic aspects:* despite the fact that we do not compute arguments and attacks in a brute-force way, optimization techniques are

[1] Users who received meta-level dialectical explanations.

needed to facilitate reasoning in this kind of argumentation frameworks. Thanks to the existential rules framework, arguments computation can be done on the fly in a backward manner using query rewriting techniques such as König et al. (2015). Attacks computation can be done using an intermediate compilation step where a hypergraph of conflicts is constructed (Chomicki et al., 2004) and then attacks are computed based on this hypergraph. Optimization techniques are needed to compute such hypergraph using heuristics that are based on inconsistency measures (Hunter and Konieczny, 2005).

- *Redundancy:* when computing arguments we do not consider equivalent arguments. Taking into account equivalence between arguments would reduce redundancy, consequently making the computational process less hard. Moreover, the interaction with the user would be more sense-making.

One-shot Argument-based Explanations. In Chapter 3 we characterized universal acceptance using proponent sets and blocks. We propose the following directions for future work in a prioritized order:

- *Relation with causality in databases:* in Meliou et al. (2010) causality and responsibility have been used to provide a model of explanation within the framework of Halpern and Pearl (2005) for query answering in *consistent databases*. Causes are assigned degrees of responsibility assigned to them based on their contributions. The idea is to investigate how the concept of block and proponent set can be modeled in the Halpern-Pearl's framework. Moreover, investigating the notion of responsibility would help in putting preferences between explanations (Chockler and Halpern, 2004).

- *Relation with BAF:* blocks and proponent sets seem to have a relation with Bipolar Argumentation Frameworks (BAF) of Cayrol and Lagasquie-Schiex (2005). Since the notion of supporting argument can be defined between an argument and a query, one can generalize it to cover support between arguments. If a correspondence is found then the dialectical proof theory proposed in this thesis could be of use in Bipolar Argumentation Frameworks.

- *Characterization of other semantics:* in Chapter 1, Section 1.3 we have presented a list of inconsistency-handling mechanism (e.g. Noobjection, Lazy, etc.). Investigating such semantics with respect to argumentation semantics would reveal interesting results.

Meta-level Dialectical Explanations. In Chapter 4 we provided a dialectical proof theory that computes (non-)universal acceptance and whose proofs are called Meta-level Dialectical Explanations. This work can be further investigated in the following order:

- *Computational complexity:* no computational complexity results have been provided as it was not the main focus of the thesis. However, investigating such issue would shed light on many aspects, especially algorithmic ones.

- *Dialectical proofs generation:* when it comes to explanation in general we often look for short explanations. The dispute complexity studied in this contribution can be combined with heuristics to provide short proofs, consequently short explanations.

- *Dialectical proofs for other semantics:* this would be the result of the last point of future work for One-shot Argument-based Explanations. Providing dialectical proof theories for other semantics would contribute to the improvement of their usability.

Object-level Dialectical Explanations. In Chapter 5 we provided a dialogue model of explanation that handles the content-sensitivity of explanations. The future directions are as follows with a prioritized order:

- *Experimentation:* in the thesis, we provided a preliminary evaluation of Object-level Dialectical Explanations in form of a use-case. Scaling up the experiment would give more conclusive results.

- *General semantics of explanation dialogues:* the dialogue model proposed in this thesis was an instantiation of the one provided in Arioua and Croitoru (2015); Walton (2011). For both, our and Walton's model, a general semantics is needed to broaden the use of such dialogues, especially in Multi-Agent Systems. The biggest obstacle, we believe, would be the formal account of understanding. A interesting and pragmatical starting point would be to consider the knowledge account of understanding (Grimm, 2006), which has been addressed by to many philosophers such as Achinstein, Kitcher, Lipton, Salmon, and Woodward.[2] This view states that understanding is a species of knowledge, more precisely causal knowledge (Lipton, 2003). In such case the usual possible worlds semantics can be used to represent different understanding states.

- *Extending the locutions:* in the dialogue model, the EXPLAIN locution is understood as a why question. Extending the set of locutions to accommodate other explanatory locutions as "What is?", "How?", "How would be?" would improve the expressiveness of the dialogue model and allow more usability functions.

DALEK and the Durum Wheat KB: we have provided an implementation of dialectical explanations. This implementation should be evaluated with respect to usability through direct interactions with the users. This urges the need to develop a natural language translator from/to First-order Logic. The interesting thing in the existential rules framework is that if the underlaying knowledge base is authored in the Conceptual Graph framework then the translation to natural language would be straightforward as CG are proposed in the first place for this particular problem. The translation is done by considering a subset of the English language called controlled English (e.g. Kuhn (2014)) which can provide minimal linguistics functionalities for DALEK. Moreover, the Durum Wheat knowledge base can be improved by integrating more multi-disciplinary knowledge from pasta transformation and socio-economical knowledge. This would give more material for analyzing explanation dialogues over real-world knowledge bases.

[2] A very useful discussion on the recent philosophical trends on understanding and explanation can be found in (de Regt, 2013).

8
Appendix

8.1 Chapter 3

Example 8.1.1 (Pick two!)**.** *Table 8.2 contains all arguments of Example 3.3.4, page 53.*

$a_1 = (\{project(P), isfast(P), ischeap(P)\}, project(P) \wedge isfast(P) \wedge ischeap(P))$
$a_2 = (\{project(P), isfast(P), isgood(P)\}, project(P) \wedge isfast(P) \wedge isgood(P))$
$a_3 = (\{project(P), isgood(P), ischeap(P)\}, project(P) \wedge isgood(P) \wedge ischeap(P))$
$a_4 = (\{project(P), isfast(P), ischeap(P)\}, project(P) \wedge isfast(P))$
$a_5 = (\{project(P), isfast(P), ischeap(P)\}, project(P) \wedge ischeap(P))$
$a_6 = (\{project(P), isfast(P), ischeap(P)\}, isfast(P) \wedge ischeap(P))$
$a_7 = (\{project(P), isgood(P), ischeap(P)\}, project(P) \wedge isgood(P))$
$a_8 = (\{project(P), isgood(P), ischeap(P)\}, project(P) \wedge ischeap(P))$
$a_9 = (\{project(P), isgood(P), ischeap(P)\}, isgood(P) \wedge ischeap(P))$
$a_{10} = (\{project(P), isfast(P), isgood(P)\}, project(P) \wedge isfast(P))$
$a_{11} = (\{project(P), isfast(P), isgood(P)\}, project(P) \wedge isgood(P))$
$a_{12} = (\{project(P), isfast(P), isgood(P)\}, isfast(P) \wedge isgood(P))$
$a_{13} = (\{project(P), isfast(P), isgood(P)\}, project(P))$
$a_{14} = (\{project(P), isfast(P), isgood(P)\}, isfast(P))$
$a_{15} = (\{project(P), isfast(P), isgood(P)\}, isgood(P))$
$a_{16} = (\{project(P), isgood(P), ischeap(P)\}, project(P))$
$a_{17} = (\{project(P), isgood(P), ischeap(P)\}, isgood(P))$
$a_{18} = (\{project(P), isgood(P), ischeap(P)\}, ischeap(P))$
$a_{19} = (\{project(P), isfast(P), ischeap(P)\}, project(P))$
$a_{20} = (\{project(P), isfast(P), ischeap(P)\}, isfast(P))$
$a_{21} = (\{project(P), isfast(P), ischeap(P)\}, ischeap(P))$
$a_{22} = (\{project(P), isfast(P)\}, project(P) \wedge isfast(P))$
$a_{23} = (\{project(P), isgood(P)\}, project(P) \wedge isgood(P))$
$a_{24} = (\{isgood(P), ischeap(P)\}, isgood(P) \wedge ischeap(P))$
$a_{25} = (\{project(P), ischeap(P)\}, project(P) \wedge ischeap(P))$
$a_{26} = (\{isfast(P), ischeap(P)\}, isfast(P) \wedge ischeap(P))$
$a_{27} = (\{isfast(P), isgood(P)\}, isfast(P) \wedge isgood(P))$
$a_{28} = (\{project(P), isgood(P)\}, project(P))$
$a_{29} = (\{project(P), isgood(P)\}, isgood(P))$
$a_{30} = (\{isgood(P), ischeap(P)\}, isgood(P))$
$a_{31} = (\{isgood(P), ischeap(P)\}, ischeap(P))$
$a_{32} = (\{project(P), isfast(P)\}, project(P))$
$a_{33} = (\{project(P), isfast(P)\}, isfast(P))$
$a_{34} = (\{project(P), ischeap(P)\}, project(P))$
$a_{35} = (\{project(P), ischeap(P)\}, ischeap(P))$
$a_{36} = (\{isfast(P), ischeap(P)\}, isfast(P))$
$a_{37} = (\{isfast(P), ischeap(P)\}, ischeap(P))$
$a_{38} = (\{isfast(P), isgood(P)\}, isfast(P))$
$a_{39} = (\{isfast(P), isgood(P)\}, isgood(P))$
$a_{40} = (\{project(P)\}, project(P))$
$a_{41} = (\{isfast(P)\}, isfast(P))$
$a_{42} = (\{ischeap(P)\}, ischeap(P))$
$a_{43} = (\{isgood(P)\}, isgood(P))$
$a_{44} = (\{isfast(P)\}, isfast(P))$
$a_{45} = (\{project(P), isfast(P), isgood(P)\}, project(P) \wedge isfast(P) \wedge isgood(P) \wedge isexpensive(P))$

Table 8.1: Arguments from 1 to 45.

$a_{46} = (\{project(P), isfast(P), isgood(P)\}, isfast(P) \wedge isgood(P) \wedge isexpensive(P))$
$a_{47} = (\{project(P), isfast(P), isgood(P)\}, project(P) \wedge isexpensive(P))$
$a_{48} = (\{project(P), isfast(P), isgood(P)\}, isfast(P) \wedge isexpensive(P))$
$a_{49} = (\{project(P), isfast(P), isgood(P)\}, isgood(P) \wedge isexpensive(P))$
$a_{50} = (\{isfast(P), isgood(P)\}, isfast(P) \wedge isgood(P) \wedge isexpensive(P))$
$a_{51} = (\{project(P), isfast(P), isgood(P)\}, isexpensive(P))$
$a_{52} = (\{isfast(P), isgood(P)\}, isfast(P) \wedge isexpensive(P))$
$a_{53} = (\{isfast(P), isgood(P)\}, isgood(P) \wedge isexpensive(P))$
$a_{54} = (\{isfast(P), isgood(P)\}, isexpensive(P))$

Table 8.2: Arguments from 46 to 54.

Bibliography

Abiteboul, Serge, Richard Hull, and Victor Vianu (1995), *Foundations of databases*. Addison-Wesley Reading.

Amgoud, Leila (2014), "Postulates for logic-based argumentation systems." *International Journal of Approximate Reasoning*, 55, 2028–2048.

Amgoud, Leila, Sihem Belabbès, and Henri Prade (2006), "A formal general setting for dialogue protocols." In *Artificial Intelligence: Methodology, Systems, and Applications*, 13–23, Springer.

Amgoud, Leila and Philippe Besnard (2009), "Bridging the gap between abstract argumentation systems and logic." In *Proceedings of the International Conference on Scalable Uncertainty Management (SUM'2009)*, 12–27, Springer.

Amgoud, Leila and Philippe Besnard (2010), "A formal analysis of logic-based argumentation systems." In *Proceedings of the International Conference on Scalable Uncertainty Management (SUM'10)*, 42–55, Springer.

Amgoud, Leila and Philippe Besnard (2013), "Logical limits of abstract argumentation frameworks." *Journal of Applied Non-Classical Logics*, 23, 229–267.

Amgoud, Leila and Claudette Cayrol (1998), "On the acceptability of arguments in preference-based argumentation." In *Proceedings of the International Conference on Uncertainty in Artificial Intelligence (UAI'98)*, 1–7, Morgan Kaufmann Publishers Inc.

Amgoud, Leila, Yannis Dimopoulos, and Pavlos Moraitis (2008), "Making decisions through preference-based argumentation." *Proceedings of the International Conference on Principles of Knowledge Representation and Reasoning (KR'08)*, 8, 963–970.

Amgoud, Leila, De Saint-Cyr, and Florence Dupin (2013), "An axiomatic approach for persuasion dialogs." In *Proceedings of IEEE International Conference on Tools with Artificial Intelligence (ICTAI'13)*, 618–625, IEEE.

Arenas, Marcelo, Leopoldo Bertossi, and Jan Chomicki (1999), "Consistent query answers in inconsistent databases." In *Proceedings of the eighteenth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, 68–79, ACM.

Arioua, Abdallah and Madalina Croitoru (2015), "Formalizing explanatory dialogues." In *Proceedings of the 9th International Conference on Scalable Uncertainty Management (SUM'15)*, 282–297.

Arioua, Abdallah and Madalina Croitoru (2016a), "Dalek: a tool for dialectical explanations in inconsistent knowledge bases." In *Proceedings of the 6th International Conference on Computational Models of Argument, COMMA 2016*.

Arioua, Abdallah and Madalina Croitoru (2016b), "Dialectical characterization of consistent query explanation with existential rules." In *Proceedings of the 29th International Florida Artificial Intelligence Research Society Conference (FLAIRS'16)*, 14–19.

Arioua, Abdallah and Madalina Croitoru (2016c), "A dialectical proof theory for universal acceptance in coherent logic-based argumentation frameworks." In *Proceedings of the 22nd European Conference on Artificial Intelligence, (ECAI'16)*.

Arioua, Abdallah, Nouredine Tamani, and Madalina Croitoru (2014a), "On conceptual graphs and explanation of query answering under inconsistency." In *Proceedings of the 21st International Conference on Conceptual Structures (ICCS'2014)*, 51–64.

Arioua, Abdallah, Nouredine Tamani, and Madalina Croitoru (2015a), "Query answering explanation in inconsistent datalog +/- knowledge bases." In *Proceedings of the 26th International Conference Database and Expert Systems Applications (DEXA'15), Part I*, 203–219.

Arioua, Abdallah, Nouredine Tamani, Madalina Croitoru, and Patrice Buche (2014b), "Query failure explanation in inconsistent knowledge bases: a dialogical approach." In *Research and Development in Intelligent Systems XXXI, SGAI 2014*, 119–133, Springer International Publishing.

Arioua, Abdallah, Nouredine Tamani, Madalina Croitoru, and Patrice Buche (2014c), "Query failure explanation in inconsistent knowledge bases using argumentation." In *Proceedings of Computational Models of Argument (COMMA'14)*, 101–108.

Arioua, Abdallah, Nouredine Tamani, Madalina Croitoru, Jérôme Fortin, and Patrice Buche (2015b), "Investigating the mapping between default logic and inconsistency-tolerant semantics." In *Proceedings of the 14th*

International Conference on Artificial Intelligence and Soft Computing (ICAISC'15), Part II, 554–564.

Atkinson, Katie, Trevor Bench-Capon, and Peter McBurney (2005), "A dialogue game protocol for multi-agent argument over proposals for action." *Journal of Autonomous Agents and Multi-Agent Systems*, 11, 153–171.

Austin, John Langshaw (1975), *How to do things with words*. Oxford university press.

Baader, Franz, Sebastian Brandt, and Carsten Lutz (2005), "Pushing the el envelope." In *Proceedings of the 19th International Joint Conference on Artificial Intelligence (IJCAI'05)*.

Baader, Franz, Rafael Penaloza, and Boontawee Suntisrivaraporn (2007), "Pinpointing in the description logic el+." In *Annual Conference on Artificial Intelligence*, 52–67, Springer.

Baget, Jean-François, Marie-Laure Mugnier, Sebastian Rudolph, and Michaël Thomazo (2011a), "Walking the complexity lines for generalized guarded existential rules." In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI'11)*, 712–717.

Baget, Jean-François, Salem Benferhat, Zied Bouraoui, Madalina Croitoru, Marie-Laure Mugnier, Odile Papini, Swan Rocher, and Karim Tabia (2016), "A general modifier-based framework for inconsistency-tolerant query answering." In *Proceedings of the Fifteenth International Conference on Principles of Knowledge Representation and Reasoning (KR'16)*, 513–516.

Baget, Jean-François, Michel Leclère, Marie-Laure Mugnier, Swan Rocher, and Clément Sipieter (2015), "Graal: A toolkit for query answering with existential rules." In *Rule Technologies: Foundations, Tools, and Applications - 9th International Symposium, RuleML 2015, Proceedings*, 328–344.

Baget, Jean-Franois, Michel Leclre, Marie-Laure Mugnier, and Eric Salvat (2011b), "On rules with existential variables: Walking the decidability line." *Artificial Intelligence*, 175, 1620 – 1654.

Baroni, Pietro and Massimiliano Giacomin (2009), "Semantics of abstract argument systems." In *Argumentation in artificial intelligence*, 25–44, Springer.

Beeri, Catriel and Moshe Y Vardi (1981), “The implication problem for data dependencies.” In *International Colloquium on Automata, Languages, and Programming*, 73–85, Springer.

Benferhat, Salem, Didier Dubois, and Henri Prade (1993), “Argumentative inference in uncertain and inconsistent knowledge bases.” In *Proceedings of the Ninth International Conference on Uncertainty in Artificial Intelligence*, 411–419, Morgan Kaufmann Publishers Inc.

Benferhat, Salem, Didier Dubois, and Henri Prade (1995), “A local approach to reasoning under inconsistency in stratified knowledge bases.” In *Proceedings of the European Conference on Symbolic and Quantitative Approaches to Reasoning and Uncertainty (ECSQARU’95)*, 36–43, Springer.

Bertossi, Leopoldo (2006), “Consistent query answering in databases.” *ACM Sigmod Record*, 35, 68–76.

Bertossi, Leopoldo (2011), “Database repairing and consistent query answering.” *Synthesis Lectures on Data Management*, 3, 1–121.

Bertossi, Leopoldo, Anthony Hunter, and Torsten Schaub (2005), “Introduction to inconsistency tolerance.” In *Inconsistency Tolerance*, 1–14, Springer.

Besnard, Philippe and Anthony Hunter (2008), *Elements of argumentation*, volume 47. MIT press Cambridge.

Besnard, Philippe and Anthony Hunter (2014), “Constructing argument graphs with deductive arguments: a tutorial.” *Argument & Computation*, 5, 5–30.

Bienvenu, Meghyn (2012), “On the complexity of consistent query answering in the presence of simple ontologies.” In *Proceedings of the Twenty-Sixth AAAI Conference on Artificial Intelligence.*

Bienvenu, Meghyn, Camille Bourgaux, and François Goasdoué (2014), “Querying inconsistent description logic knowledge bases under preferred repair semantics.” In *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence*, 996–1002.

Bienvenu, Meghyn, Camille Bourgaux, and François Goasdoué (2016), “Explaining inconsistency-tolerant query answering over description logic knowledge bases." In *Proceedings of the 13th AAAI Conference on Artificial Intelligence*, 900–906.

Bienvenu, Meghyn and Riccardo Rosati (2013), "Tractable approximations of consistent query answering for robust ontology-based data access." In *Proceedings of the 23rd International Joint Conference on Artificial Intelligence (IJCAI'13)*.

Bondarenko, Andrei, Francesca Toni, and Robert A Kowalski (1993), "An assumption-based framework for non-monotonic reasoning." In *Proceedings of the International Conference on Logic Programming and Non-monotonic Reasoning (LPNMR'93)*, volume 93, 171–189.

Borgida, Alexander, Enrico Franconi, Ian Horrocks, Deborah L McGuinness, and Peter F Patel-Schneider (2000), "Explaining alc subsumption." In *Proceedings of the European Conference on Artificial Intelligence (ECAI 2000)*, 209–213.

Bouraoui, Zied, Salem Benferhat, Madalina Croitoru, Odile Papini, and Karim Tabia (2016), "On the use of non-objection inference in inconsistent lightweight ontologies." In *Proceedings of the 25th International Joint Conference on Artificial Intelligence (IJCAI'16)*.

Calı, Andrea, Georg Gottlob, and Michael Kifer (2008), "Taming the infinite chase: Query answering under expressive relational constraints." *Proceedings of the International Conference on Principles of Knowledge Representation and Reasoning (KR'08)*, 70–80.

Calì, Andrea, Georg Gottlob, and Thomas Lukasiewicz (2009a), "Datalog$^{\pm}$: a unified approach to ontologies and integrity constraints." In *Proceedings of the 12th International Conference on Database Theory (ICDT'09)*, 14–30.

Calì, Andrea, Georg Gottlob, and Thomas Lukasiewicz (2009b), "A general datalog-based framework for tractable query answering over ontologies." In *Proceedings of the Twenty-Eigth ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems (PODS'09)*, 77–86.

Calì, Andrea, Georg Gottlob, and Thomas Lukasiewicz (2012), "A general datalog-based framework for tractable query answering over ontologies." *Journal of Web Semantics*, 14, 57–83.

Cali, Andrea, Georg Gottlob, Thomas Lukasiewicz, Bruno Marnette, and Andreas Pieris (2010), "Datalog+/-: A family of logical knowledge representation and query languages for new applications." In *Proceedings of the 25th Annual IEEE Symposium on Logic in Computer Science (LICS'2010)*, 228–242, IEEE.

Calvanese, Diego, Giuseppe De Giacomo, Domenico Lembo, Maurizio Lenzerini, and Riccardo Rosati (2005), "Inconsistency tolerance in p2p data integration: an epistemic logic approach." In *International Workshop on Database Programming Languages*, 90–105, Springer.

Calvanese, Diego, Giuseppe De Giacomo, Domenico Lembo, Maurizio Lenzerini, and Riccardo Rosati (2007), "Tractable reasoning and efficient query answering in description logics: The dl-lite family." *Journal of Automated Reasoning*, 39, 385–429.

Calvanese, Diego, Magdalena Ortiz, Mantas Simkus, and Giorgio Stefanoni (2013), "Reasoning about explanations for negative query answers in dl-lite." *Journal of Artificial Intelligence Research*, 635–669.

Caminada, Martin (2006), "On the issue of reinstatement in argumentation." In *Logics in artificial intelligence*, 111–123, Springer.

Caminada, Martin and Leila Amgoud (2005), "An axiomatic account of formal argumentation." In *Proceedings of the Twentieth National Conference on Artificial Intelligence and the Seventeenth Innovative Applications of Artificial Intelligence Conference (AAAI'05)*, 608–613.

Caminada, Martin and Leila Amgoud (2007), "On the evaluation of argumentation formalisms." *Artificial Intelligence*, 171 (5-6), 286–310.

Cayrol, Claudette (1995), "On the relation between argumentation and non-monotonic coherence-based entailment." In *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence, IJCAI 95*, 1443–1448.

Cayrol, Claudette, Sylvie Doutre, and Jérôme Mengin (2003), "On decision problems related to the preferred semantics for argumentation frameworks." *Journal of logic and computation*, 13, 377–403.

Cayrol, Claudette and Marie-Christine Lagasquie-Schiex (2005), "On the acceptability of arguments in bipolar argumentation frameworks." In *Proceedings of the European Conference on Symbolic and Quantitative Approaches to Reasoning and Uncertainty*, 378–389, Springer.

Ceri, Stefano, Georg Gottlob, and Letizia Tanca (1989), "What you always wanted to know about datalog (and never dared to ask)." *IEEE Transactions on Knowledge and Data Engineering*, 1, 146–166.

Ceri, Stefano, Georg Gottlob, and Letizia Tanca (2012), *Logic programming and databases*. Springer Science & Business Media.

Chein, Michel and Marie-Laure Mugnier (2009), *Graph-based Knowledge Representation - Computational Foundations of Conceptual Graphs*. Advanced Information and Knowledge Processing, Springer.

Chockler, Hana and Joseph Y Halpern (2004), "Responsibility and blame: A structural-model approach." *Journal of Artificial Intelligence Research*, 22, 93–115.

Chomicki, Jan (2007), "Consistent query answering: Five easy pieces." In *International Conference on Database Theory (ICDT'07)*, 1–17, Springer.

Chomicki, Jan, Jerzy Marcinkowski, and Slawomir Staworko (2004), "Computing consistent query answers using conflict hypergraphs." In *Proceedings of the 2004 ACM CIKM International Conference on Information and Knowledge Management*, 417–426.

Coste-Marquis, Sylvie, Caroline Devred, and Pierre Marquis (2005), "Symmetric argumentation frameworks." In *Proceedings of the European Conference on Symbolic and Quantitative Approaches to Reasoning with Uncertainty (ECSQARU'05)*, 317–328, Springer.

Croitoru, Madalina and Srdjan Vesic (2013), "What can argumentation do for inconsistent ontology query answering?" In *Proceedings of the International Conference on Scalable Uncertainty Management (SUM'13)*, 15–29, Springer.

de Regt, Henk W (2013), "Understanding and explanation: Living apart together?" *Studies in History and Philosophy of Science Part A*, 44, 505–509.

Deutsch, Alin, Alan Nash, and Jeff Remmel (2008), "The chase revisited." In *Proceedings of the twenty-seventh ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, 149–158, ACM.

Dimopoulos, Yannis, Bernhard Nebel, and Francesca Toni (1999), "Preferred arguments are harder to compute than stable extensions." In *Pro-*

ceddings of the International Joint Conference on Artificial Intelligence (IJCAI'99), volume 16, 36–43.

Doutre, Sylvie (2002), *Autour de la sémantique préférée des systèmes d'argumentation*. Ph.D. thesis, Toulouse 3.

Doutre, Sylvie and Jérôme Mengin (2004), "On sceptical versus credulous acceptance for abstract argument systems." In *Logics in Artificial Intelligence*, 462–473, Springer.

Du, Jianfeng, Kewen Wang, and Yi-Dong Shen (2014), "A tractable approach to abox abduction over description logic ontologies." In *Proceedings of the International Conference on Artificial Intelligence (AAAI'14)*, 1034–1040.

Dung, Phan Minh (1995), "On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games." *Artificial intelligence*, 77, 321–357.

Dunne, Paul E and Trevor JM Bench-Capon (2002), "Coherence in finite argument systems." *Artificial Intelligence*, 141, 187–203.

Dunne, Paul E and Trevor JM Bench-Capon (2003), "Two party immediate response disputes: properties and efficiency." *Artificial Intelligence*, 149, 221–250.

Dunne, Paul E. and Michael Wooldridge (2009), *Argumentation in Artificial Intelligence*, chapter Complexity of Abstract Argumentation, 85–104. Springer US.

Eiter, Thomas and Georg Gottlob (1995), "The complexity of logic-based abduction." *Journal of the ACM*, 42, 3–42.

Eiter, Thomas and Georg Gottlob (2002), "Hypergraph transversal computation and related problems in logic and ai." In *European Workshop on Logics in Artificial Intelligence*, 549–564, Springer.

Eustis, Alexander (2013), *Hypergraph Independence Numbers*. Thesis, Uuniversity of California, San Diego.

Fagin, Ronald (2009), *Encyclopedia of Database Systems*, chapter Tuple-Generating Dependencies, 3201–3202. Springer US.

Falappa, Marcelo A, Gabriele Kern-Isberner, and Guillermo R Simari (2002), "Explanations, belief revision and defeasible reasoning." *Artificial Intelligence*, 141, 1–28.

Gallaire, Herve and Jean-Marie Nicolas (1987), "Logic approach to knowledge and data bases at ecrc." *IEEE Data Eng. Bull.*, 10, 2–9.

García, Alejandro J and Guillermo R Simari (2004), "Defeasible logic programming: An argumentative approach." *Theory and practice of logic programming*, 4, 95–138.

Gorogiannis, Nikos and Anthony Hunter (2011), "Instantiating abstract argumentation with classical logic arguments: Postulates and properties." *Artificial Intelligence*, 175, 1479 – 1497.

Grimm, Stephen R. (2006), "Is understanding a species of knowledge?" *British Journal for the Philosophy of Science*, 57, 515–535.

Halpern, Joseph Y and Judea Pearl (2005), "Causes and explanations: A structural-model approach. part i: Causes." *The British journal for the philosophy of science*, 56, 843–887.

Hamblin, C.L. (1970), *Fallacies.* University paperback, Methuen.

Hempel, Carl Gustav (1965), "Aspects of scientific explanation." In *Philosophy and Phenomenological Research*, 504, Free Press.

Horridge, Matthew, Bijan Parsia, and Ulrike Sattler (2010), "Justification oriented proofs in owl." In *International Semantic Web Conference (ISWC'10)*, 354–369, Springer.

Hunter, Anthony (2015), "Modelling the persuadee in asymmetric argumentation dialogues for persuasion." In *Proceedings of the 24th International Joint Conference on Artificial Intelligence (IJCAI'15)*, 3055–3061.

Hunter, Anthony and Sébastien Konieczny (2005), "Approaches to measuring inconsistent information." In *Inconsistency Tolerance*, 191–236.

Jakobovits, Hadassa and Dirk Vermeir (1999), "Dialectic semantics for argumentation frameworks." In *Proceedings of the 7th International Conference on Artificial Intelligence and Law*, 53–62, ACM.

Johnson, Mark W, Peter McBurney, and Simon Parsons (2003), "When are two protocols the same?" In *Communication in Multiagent Systems*, 253–268, Springer.

Kelley, John F (1984), "An iterative design methodology for user-friendly natural language office information applications." *ACM Transactions on Information Systems (TOIS)*, 2, 26–41.

König, Mélanie, Michel Leclère, Marie-Laure Mugnier, and Michaël Thomazo (2015), "Sound, complete and minimal ucq-rewriting for existential rules." *Journal of Semantic Web*, 6, 451–475.

Kuhn, Tobias (2014), "A survey and classification of controlled natural languages." *Computational Linguistics*, 40, 121–170.

Leclère, Michel, Marie-Laure Mugnier, and Swan Rocher (2013), "Kiabora: an analyzer of existential rule bases." In *Proceedings of the International Conference on Web Reasoning and Rule Systems (RR'13)*, 241–246, Springer.

Lembo, Domenico, Maurizio Lenzerini, Riccardo Rosati, Marco Ruzzi, and Domenico Fabio Savo (2010), "Inconsistency-tolerant semantics for description logics." In *Proceedings of the International Conference on Web Reasoning and Rule Systems (RR'10)*, 103–117, Springer-Verlag.

Levesque, Hector J. and Ronald J. Brachman (1987), "Expressiveness and tractability in knowledge representation and reasoning." *Computational Intelligence*, 3, 78–93.

Lipton, Peter (2003), *Inference to the best explanation*. Routledge.

Lorenz, Kuno (2001), "Basic objectives of dialogue logic in historical perspective." *Synthese*, 127, 255–263.

Lukasiewicz, Thomas, Maria Vanina Martinez, and Gerardo I. Simari (2012), "Inconsistency handling in datalog+/- ontologies." In *Proceedings of the 20th European Conference on Artificial Intelligence (ECAI'12)*, 558–563.

Mackenzie, Jim D (1979), "Question-begging in non-cumulative systems." *Journal of philosophical logic*, 8, 117–133.

Marnette, Bruno (2009), "Generalized schema-mappings: from termination to tractability." In *Proceedings of the twenty-eighth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, 13–22, ACM.

Martinez, Maria Vanina, Cristhian Ariel David Deagustini, Marcelo A Falappa, and Guillermo Ricardo Simari (2014), "Inconsistency-tolerant reasoning in datalog+/- ontologies via an argumentative semantics." In *Advances in Artificial Intelligence–IBERAMIA 2014*, 15–27, Springer.

Martinez, M.V., C. Molinaro, V.S. Subrahmanian, and L. Amgoud (2013), *A General Framework for Reasoning On Inconsistency*. SpringerBriefs in Computer Science, Springer New York.

McBurney, Peter and Simon Parsons (2009), "Dialogue games for agent argumentation." In *Argumentation in Artificial Intelligence* (Guillermo Simari and Iyad Rahwan, eds.), 261–280, Springer US.

McBurney, Peter, Simon Parsons, and Michael Wooldridge (2002), "Desiderata for agent argumentation protocols." In *Proceedings of the 1st International Joint Conference on Autonomous Agents and Multiagent Systems: part 1*, 402–409, ACM.

McGuinness, Deborah L and Alexander Borgida (1995), "Explaining subsumption in description logics." In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI'95)*, 816–821.

McGuinness, Deborah L. and Peter F. Patel-Schneider (1998), "Usability issues in knowledge representation systems." In *Proceedings of the Fifteenth National Conference on Artificial Intelligence (AAAI'98)*, 608–614.

Meliou, Alexandra, Wolfgang Gatterbauer, Joseph Y Halpern, Christoph Koch, Katherine F Moore, and Dan Suciu (2010), "Causality in databases." *IEEE Data Eng. Bull.*, 33, 59–67.

Meliou, Alexandra, Sudeepa Roy, and Dan Suciu (2014), "Causality and explanations in databases." *Proceedings of the VLDB Endowment*, 7, 1715–1716.

Modgil, Sanjay and Martin Caminada (2009a), *Argumentation in Artificial Intelligence*, chapter The added value of argumentation, 357–403. Springer US.

Modgil, Sanjay and Martin Caminada (2009b), *Argumentation in Artificial Intelligence*, chapter Proof Theories and Algorithms for Abstract Argumentation Frameworks, 105–129. Springer US.

Modgil, Sanjay and Henry Prakken (2013), "A general account of argumentation with preferences." *Artificial Intelligence*, 195, 361–397.

Mugnier, Marie-Laure and Michaël Thomazo (2014), "An introduction to ontology-based query answering with existential rules." In *Reasoning Web. Reasoning on the Web in the Big Data Era - 10th International Summer School 2014. Proceedings*, 245–278.

Parsons, Simon, Michael Wooldridge, and Leila Amgoud (2003), "Properties and complexity of some formal inter-agent dialogues." *Journal of Logic and Computation*, 13, 347–376.

Pitt, Joseph C. (1988), *Theories of explanation*. Oxford University Press.

Poggi, Antonella, Domenico Lembo, Diego Calvanese, Giuseppe De Giacomo, Maurizio Lenzerini, and Riccardo Rosati (2008), "Linking data to ontologies." In *Journal on data semantics X*, 133–173, Springer.

Prakken, Henry (2001), "Relating protocols for dynamic dispute with logics for defeasible argumentation." *Synthese*, 127, 187–219.

Prakken, Henry (2005), "Coherence and flexibility in dialogue games for argumentation." *Journal of Logic and Computation*, 15, 1009–1040.

Prakken, Henry (2006), "Formal systems for persuasion dialogue." *Knowledge Engineering Review*, 21, 163–188.

Prakken, Henry (2009), *Argumentation in Artificial Intelligence*, chapter Models of Persuasion Dialogue, 281–300. Springer US.

Priest, Graham (1979), "The logic of paradox." *Journal of Philosophical logic*, 8, 219–241.

Ramakrishnan, Raghu and Jeffrey D Ullman (1995), "A survey of deductive database systems." *The journal of logic programming*, 23, 125–149.

Reed, Chris and Simon Wells (2007), "Dialogical argument as an interface to complex debates." *IEEE Intelligent Systems*, 22, 60–65.

Reiter, Raymond (1980), "A logic for default reasoning." *Artificial intelligence*, 13, 81–132.

Reiter, Raymond (1987), "A theory of diagnosis from first principles." *Artificial intelligence*, 32, 57–95.

Rescher, Nicholas and Ruth Manor (1970), "On inference from inconsistent premisses." *Theory and decision*, 1, 179–217.

Schlobach, Stefan (2005), "Debugging and semantic clarification by pinpointing." In *Proceedings of the European Semantic Web Conference (ESWC 2016)*, 226–240, Springer.

Schlobach, Stefan, Ronald Cornet, et al. (2003), "Non-standard reasoning services for the debugging of description logic terminologies." In *IJCAI*, volume 3, 355–362.

Sebillotte, Michel (1978), "Itineraires techniques et evolution de la pensee agronomique." *Comptes Rendus des Séances de l'Académie d'Agriculture de France*.

Sini, Margherita and Vimlesh Yadav (2009), *Building Knowledge Models for Agropedia Indica v 1.0 Requirements, Guidelines, Suggestions.* Available at http://agropedia.iitk.ac.in/km_guidlines.pdf, visited 19-03-2015.

Sowa, John F (1976), "Conceptual graphs for a data base interface." *IBM Journal of Research and Development*, 20, 336–357.

Sowa, John F (1983), "Conceptual structures: information processing in mind and machine."

Thang, Phan Minh, Phan Minh Dung, and Nguyen Duy Hung (2009), "Towards a common framework for dialectical proof procedures in abstract argumentation." *Journal of Logic and Computation*, 19, 1071–1109.

Thunkijjanukij, Aree, Asanee Kawtrakul, Supamard Panichsakpatana, and Uamporn Veesommai (2008), "Lesson learned for ontology construction with thai rice case study." In *World conference on agricultural information and IT, IAALD AFITA WCCA 2008*, 495–502.

Vesic, S. (2013), "Identifying the class of maxi-consistent operators in argumentation." *Journal of Artificial Intelligence Research*, 47, 71–93.

Vesic, Srdjan (2011), *Preference-based argumentation frameworks: application in decision making and negotiation*. Ph.D. thesis, Université Paul Sabatier-Toulouse III.

Vreeswik, Gerard AW and Henry Prakken (2000), "Credulous and sceptical argument games for preferred semantics." In *Logics in Artificial Intelligence*, 239–253, Springer.

Walton, Douglas (2004), "A new dialectical theory of explanation." *Philosophical Explorations*, 7, 71–89.

Walton, Douglas (2006), "Examination dialogue: An argumentation framework for critically questioning an expert opinion." *Journal of Pragmatics*, 38, 745–777.

Walton, Douglas (2007), "Dialogical models of explanation." *Proceedings of the AAAI Workshop on Explanation-Aware Computing (ExaCt'07)*, 1–9.

Walton, Douglas (2011), "A dialogue system specification for explanation." *Synthese*, 182, 349–374.

Walton, Douglas (2016), "A dialogue system for evaluating explanations." In *Argument Evaluation and Evidence*, 69–116, Springer.

Walton, Douglas and Erik CW Krabbe (1995), *Commitment in dialogue: Basic concepts of interpersonal reasoning*. SUNY press.

Walton, Douglas N. (1984), *Logical Dialogue-Games and Fallacies*. Monograph Collection (Matt - Pseudo).