HEINRICH HEINE
UNIVERSITÄT DÜSSELDORF

Bridging the Gap between Online Discussions and Formal Models of Argumentation

Inaugural-Dissertation

zur Erlangung des Doktorgrades
der Mathematisch-Naturwissenschaftlichen Fakultät
der Heinrich-Heine-Universität Düsseldorf

vorgelegt von

Daniel Neugebauer
aus Düsseldorf

Düsseldorf, Juli 2019

aus dem Institut für Informatik
der Heinrich-Heine-Universität Düsseldorf

Gedruckt mit der Genehmigung der
Mathematisch-Naturwissenschaftlichen Fakultät der
Heinrich-Heine-Universität Düsseldorf

Berichterstatter:

1. Prof. Dr. Jörg Rothe

2. Jun-Prof. Dr. Dorothea Baumeister

Tag der mündlichen Prüfung:
02. Juli 2019

Abstract

Formal models of argumentation in the field of artificial intelligence (AI) research were originally developed as a tool for logical inference on the basis of incomplete or inconsistent knowledge. Recently, AI argumentation formalisms are also being used to represent and evaluate real-world argumentations. Software tools that support argumentation can benefit from the semantic evaluation mechanisms that are provided by formal argumentation models, while the application to real-world scenarios improves the scope and relevance of formal models of argumentation in AI.

This research is part of the Ph.D. program "Online Participation," supported by the North Rhine-Westphalian funding scheme "Fortschrittskollegs." One of the program's goals is to improve the quality of software-supported argumentation. This thesis contributes to this improvement by strengthening the connection between formal argumentation models from AI and software-supported discussions among humans on both sides.

The first part of the contribution of this thesis is an extension of the formal model of abstract argumentation frameworks. In its basic form, an argumentation framework is given by a set of arguments and a binary attack relation on these arguments. In the proposed extended model of incomplete argumentation frameworks, individual arguments and individual attacks may be uncertain—such elements may or may not be part of the discussion. The extended model allows the representation of a wider range of scenarios that can arise in real argumentation settings, such as intermediate states in elicitation processes or the restricted knowledge of a single participant about the state of a discussion. The technical contribution of this part of the thesis is a full analysis of how the added notion of uncertainty affects the computational complexity of core reasoning tasks in abstract argumentation—namely, the verification and acceptance problems—for various evaluation semantics. Compared to the computational complexity of the respective problems for standard argumentation frameworks, for incomplete argumentation frameworks we observe a jump in complexity for most variants of the acceptance problems, but not for most variants of the verification problem.

The second part of the contribution of this thesis is the development and implementation of translations from discussion data generated by the D-BAS web tool for dialog-based argumentation to three different AI argumentation formalisms, namely, abstract argumentation frameworks, abstract dialectical frameworks, and the ASPIC$^+$ framework. The translations are proved to satisfy established quality criteria. We developed the tool DABASCO to implement these translations, which enables D-BAS to utilize the full range of existing software tools for reasoning problems in the three argumentation models. The *argument pipeline* consisting of D-BAS, DABASCO, and these tools can automatically determine whether, e.g., a statement is acceptable or whether a participant's opinion is consistent, thus improving feedback for operators and participants of discussions.

Contents

1 Introduction **1**
1.1 Bridging the Gap . . . 2
1.2 Outline of Thesis . . . 4

2 Preliminaries **5**
2.1 Foundations of Formal Logic and Inference . . . 5
2.2 Formal Argumentation Models . . . 8
2.2.1 ASPIC$^+$ . . . 8
2.2.2 Abstract Argumentation Frameworks . . . 13
2.2.3 Abstract Dialectical Frameworks . . . 17
2.3 Computational Complexity . . . 19
2.3.1 Complexity of Decision Problems in the Polynomial Hierarchy . . . 20
2.3.2 Complexity of Reasoning Problems in Abstract Argumentation . . . 25

3 Verification in Incomplete Argumentation Frameworks **29**
3.1 Summary . . . 29
3.2 Personal Contribution . . . 30

4 Credulous and Skeptical Acceptance in Incomplete Argumentation Frameworks **31**
4.1 Summary . . . 31
4.2 Personal Contribution . . . 32

5 Generating Defeasible Theories from Real-World Discussions using D-BAS **33**
5.1 Summary . . . 33
5.2 Personal Contribution . . . 34

6 Conclusions and Future Work **35**
6.1 Results . . . 35
6.2 Future Work Directions . . . 36

Chapter 1

Introduction

In recent years, the rise of the Internet opened the possibility to relocate various areas of public communication from the real world to online platforms. Online forums, comment sections on web sites, and digital questionnaires provide a low-threshold opportunity for participants to inform themselves about current debates, to convey their opinion, and to learn about opinions and arguments of other users. Operators of such online participation platforms often are interested in having an overview of which proposals were made, which opinions were most common, and which are the most important arguments for and against the proposals. This creates a demand for systems that allow to automatically evaluate the arguments that were created by users in online discussions. One common approach is to use machine learning techniques, like argument mining or opinion and sentiment mining (surveys are given by, e.g., Pang and Lee (2008) and Peldszus and Stede (2013)). However, such mechanisms typically do not provide an explanation to back their results. An alternative way that ensures explainable results is to implement evaluation tools for online participation platforms using formal argumentation models.

Within the field of artificial intelligence (AI) research, formal models of argumentation have emerged as a range of formalisms that allow solving reasoning tasks with inconsistent data or with uncertain inference schemes. In applications, this data could be, e.g., a digital agent's current information about the world's state, or the different positions and statements in an online discussion. Uncertain schemes of inference include logical induction, abduction, or the application of common-sense reasoning—each of which is highly useful in practice, but may lead to incorrect conclusions in some cases. Interesting reasoning tasks could be to automatically determine what to believe about the current state of the world, what action to take, or what argument to accept in a discussion. The interest for formal argumentation in AI was most notably initiated by the work of Pollock (1970) on defeasible reasoning within the area of philosophical logic, which was later transferred to AI by Pollock (1987). A connection between defeasible rule-based logics and argument graphs was subsequently drawn by Pollock (1994) and Dung (1995), sparking a wave of research on different models of argumentation in AI—an overview is given in a variety of surveys by, e.g., Chesñevar et al. (2000), Prakken and Vreeswijk (2001), Besnard and Hunter (2008), Carstens et al. (2015), and Prakken (2017), and the book by Rahwan and Simari (2009).

Recently, the potential of formal argumentation models to enhance online platforms was recognized by AI researchers—see, e.g., the work by Scheuer et al. (2010), Heras et al. (2010), Snaith et al. (2010), Toni and Torroni (2011), Bex et al. (2013), and Wyner et al. (2016). In order to utilize formal models of argumentation in actual applications, an evaluation pipeline must be established which starts with the argumentation data to be evaluated and outputs recommendations on what to believe or what action to take. There are many possible ways to establish such a pipeline. Caminada and Amgoud (2007) propose a pipeline that uses defeasible theories as defined in Chapter 2 of this thesis as an intermediate representation of real-world data, which are then used to instantiate formal argumentation models. The following description embeds the four steps of their pipeline in a six-step process that, in addition, explicitly draws the connection between the formal representation of a discussion and the actual application level at both the start and the end. Figure 1 displays all steps of the pipeline and indicates their level of abstraction.

1. Create a knowledge base and a set of inference rules from the application.
2. Construct arguments from knowledge base and rules.
3. Derive dialectical relations between arguments.
4. Determine the dialectical status of the arguments from the relations.
5. Derive accepted conclusions from accepted arguments.
6. Interpret accepted conclusions on the application level.

1.1 Bridging the Gap

A majority of research on argumentation in AI concentrates on problems in the "middle" of the pipeline and is not concerned about the origin of the arguments or the interpretation of the results. The aim of this thesis is to help with bridging gaps in the argument evaluation pipeline. Our contribution is twofold.

Our first contribution is an extension of the model of abstract argumentation frameworks introduced by Dung (1995), which allows the representation of incomplete knowledge about elements in a discussion, and an analysis of the computational complexity of reasoning problems in such incomplete argumentation frameworks. Our extended model increases the scope of abstract argumentation frameworks and may allow using them in an extended range of applications, where complete knowledge is not available. Our classification of the complexity of reasoning problems in the extended model establishes bounds on the amount of time required to find an answer to a given problem. This information helps with choosing optimal ways to implement solutions to the problem. While simple problems can be consistently solved in short time by dedicated, exact algorithms, similar algorithms for harder problems might take an infeasibly long time for some instances. For such harder problems, rather than using a dedicated algorithm, it might be more suitable to implement a reduction to optimized SAT-solvers (see, e.g., Prasad et al. (2005)), or to trade accuracy for speed by implementing a fast approximation algorithm (see, e.g., Johnson (1974)). In order to generalize a reasoning problem for incomplete argumentation frameworks, we formalize a *possible* and a *necessary* variant of it, which reduce the answer to finding the solution to the original problem in some, respectively, in all, completions of the incomplete argumentation.

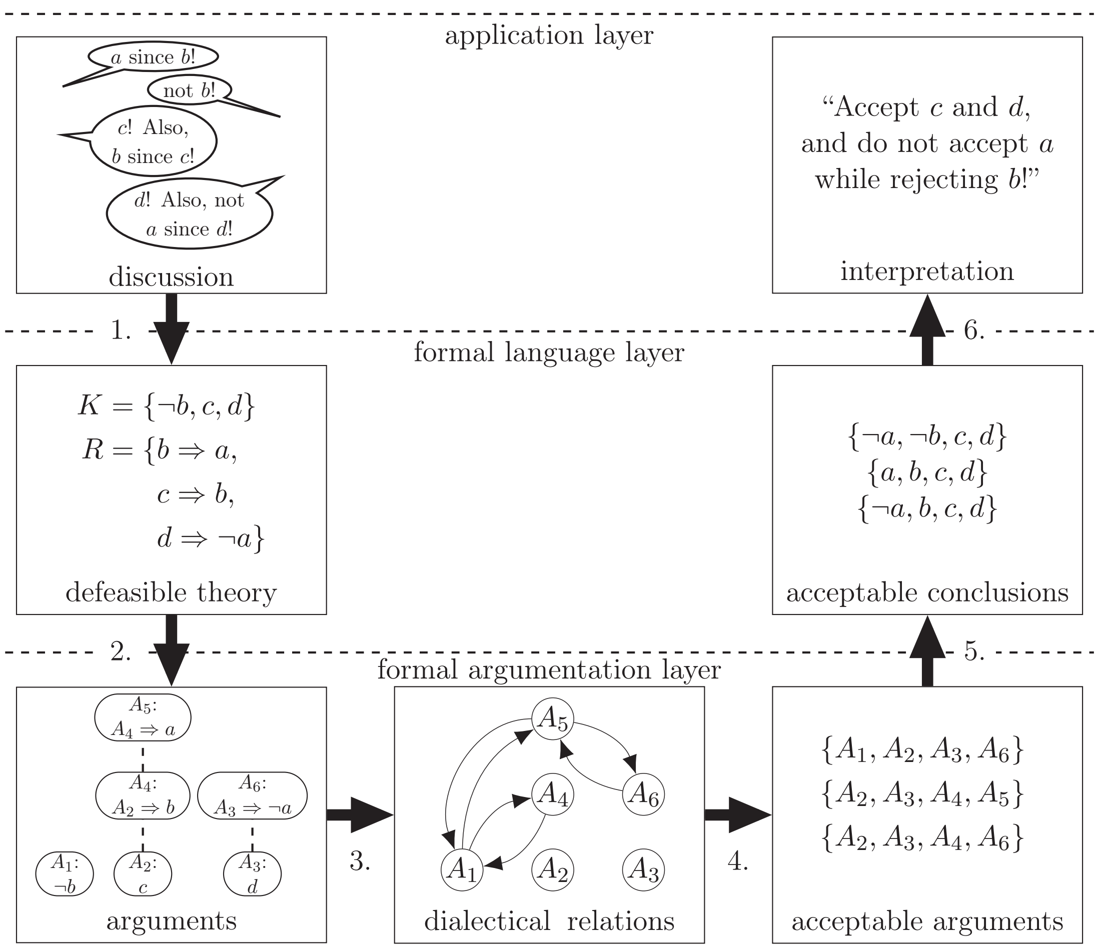


Figure 1: Visualization of the different layers of abstraction and different evaluation steps in an argumentation pipeline as proposed by Caminada and Amgoud (2007).

We newly incorporate this approach in abstract argumentation theory, which has formerly already been used by, e.g., Konczak and Lang (2005) in voting theory, by Bouveret et al. (2010) in fair division theory, by Lang et al. (2015) in algorithmic game theory, and by Baumeister et al. (2015a) in judgment aggregation theory. On the other hand, our contribution aligns with other research on abstract argumentation frameworks that also aims at increasing their domain of application, such as probabilistic and fuzzy generalizations of argumentation frameworks by Janssen et al. (2008), Dung and Thang (2010), Li et al. (2011), Rienstra (2012), and Hunter (2014), as well as various notions of dynamic change in argumentation frameworks by Boella et al. (2009), Cayrol et al. (2010), Baumann and Brewka (2010), Liao et al. (2011), Coste-Marquis et al. (2015), and Wallner et al. (2016). A broad overview of generalizations of abstract argumentation frameworks is given by Brewka et al. (2014), while Cayrol and Lagasquie-Schiex (2005) survey different forms of graduality in abstract argumentation, and Doutre and Mailly (2018) survey different concepts of dynamic updates in argumentation frameworks.

Our second contribution is the development and implementation of a software tool that automatically generates instantiations of the ASPIC$^+$ framework by Modgil and Prakken (2013), abstract argumentation frameworks by Dung (1995), and abstract dialectical frameworks by Brewka et al. (2013) from online discussions of the D-BAS platform created by Krauthoff et al. (2016, 2018). Together with a D-BAS instance and existing solver software for reasoning problems in the respective target models, our tool DABASCO provides a full argument evaluation pipeline without the need for human intervention. Other tools that support the formal representation of online discussion data include AIFdb by Lawrence et al. (2012), which is a web interface that allows to store and retrieve argumentation data in the Argument Interchange Format (AIF) by Chesñevar et al. (2006); ArguBlogging by Bex et al. (2014), which integrates discussions on online blogs into a semantic web based on the AIF; PIRIKA by Oomidou et al. (2014), which provides an implementation of the full argumentation pipeline as a standalone solution; and dAceRules by Diller et al. (2017), which allows to encode knowledge bases with strict and defeasible rules in the ACE language data format by Fuchs et al. (2008) for further evaluation. Argument mapping is a related approach that allows the structuring and displaying of argumentations with the aim to help users organize their information on a topic and to identify better conclusions. The main difference to other semi-formal argumentation tools—and in particular, to D-BAS—is that users of argument mapping tools need to be aware of and actively use the formal structure underlying the tool. Existing argument mapping tools include Araucaria by Reed and Rowe (2004), Rationale by van Gelder (2007), Carneades by Gordon et al. (2007), DebateGraph by Baldwin and Price (2008), Cohere by Shum (2008), and OVA+ by Reed et al. (2014), each of which provide a frontend for constructing structured arguments from natural language inputs. Surveys are given by Rahwan (2008), Schneider et al. (2013), and Bex et al. (2013).

1.2 Outline of Thesis

This thesis presents our contributions made towards applying formal models of argumentation for the evaluation of online discussions. Chapter 2 presents required notation, models, and fundamental results from the areas of formal logic, the ASPIC$^+$ framework, abstract argumentation frameworks, abstract dialectical frameworks, and computational complexity theory. The following Chapters present our results. Chapters 3 and 4 introduce the model of incomplete abstract argumentation frameworks and present our complete study of the computational complexity of possible and necessary variants of the VERIFICATION problem (Chapter 3) and CREDULOUS-ACCEPTANCE and SKEPTICAL-ACCEPTANCE problems (Chapter 4). Chapter 5 presents the DABASCO software tool that allows to translate D-BAS discussions to instantiations of abstract argumentation frameworks, the ASPIC$^+$ framework, and abstract dialectical frameworks. Finally, Chapter 6 summarizes the contributions of this thesis and surveys some interesting ways in which this line of research can be continued.

Chapter 2

Preliminaries

This chapter defines and illustrates all existing technical terms and models that are required for the following chapters, starting with a background in formal logic in Section 2.1, followed by introductions to the formal argumentation models used in this thesis in Section 2.2, and finally the foundations of computational complexity theory in Section 2.3.

2.1 Foundations of Formal Logic and Inference

We start by defining all notions from propositional logic and first-order logic that will be used in this thesis. A *propositional variable* x is a variable that ranges over the set of Boolean truth values $\{\texttt{true}, \texttt{false}\}$. Given a set $X = \{x_1, \ldots, x_n\}$ of propositional variables, a *truth assignment* τ_X on X is a function $\tau_X : X \to \{\texttt{true}, \texttt{false}\}$ that assigns a truth value to each variable in X. A propositional variable x has two *literals*, x and $\neg x$. Based on the syntactical negation "$\neg$" on literals, a corresponding semantic negation "$\sim$" can be obtained by, for a literal x, setting ${\sim}x = \neg x$ if x is a propositional variable, and setting ${\sim}x = x'$ if $x = \neg x'$ for some propositional variable x'. We call a set L of literals *consistent* if there are no literals $x_1, x_2 \in L$ with $x_1 = {\sim}x_2$.

Each literal is also a *propositional formula*. Further, when φ and ψ are propositional formulas, then $\neg\varphi$, $\varphi \vee \psi$, and $\varphi \wedge \psi$ are also propositional formulas. When X is the set of all propositional variables that occur in literals in a formula φ, we say that φ is a formula over X. Given a propositional formula φ over X and an assignment τ_X on X, we denote by $\varphi[\tau_X]$ the truth value that φ evaluates to under τ_X.

- If φ is a non-negated literal, then $\varphi[\tau_X] = \tau_X(\varphi)$, i.e., the formula φ evaluates to the truth value that τ_X assigns to φ;
- If $\varphi = \neg\psi$ for some formula ψ, then $\varphi[\tau_X] = \texttt{true}$ if $\psi[\tau_X] = \texttt{false}$ and $\varphi[\tau_X] = \texttt{false}$ otherwise;
- If $\varphi = \psi_1 \vee \psi_2$ for some formulas ψ_1, ψ_2, then $\varphi[\tau_X] = \texttt{true}$ if $\psi_1[\tau_X] = \texttt{true}$ or $\psi_2[\tau_X] = \texttt{true}$, and $\varphi[\tau_X] = \texttt{false}$ otherwise;

- If $\varphi = \psi_1 \wedge \psi_2$ for some formulas ψ_1, ψ_2, then $\varphi[\tau_X] =$ `true` if $\psi_1[\tau_X] =$ `true` and $\psi_2[\tau_X] =$ `true`, and $\varphi[\tau_X] =$ `false` otherwise.

An assignment τ_X on a set X of literals is a *satisfying assignment* for a formula φ if $\varphi[\tau_X]$ evaluates to `true`.

A formula φ is a *disjunction* of formulas ψ_1 through ψ_n if $\varphi = \psi_1 \vee \psi_2 \vee \cdots \vee \psi_n$, which can also be written as $\varphi = \bigvee_{i=1}^{n} \psi_i$. φ is a *conjunction* of formulas ψ_1 through ψ_n if $\varphi = \psi_1 \wedge \psi_2 \wedge \cdots \wedge \psi_n$, which is also written as $\varphi = \bigwedge_{i=1}^{n} \psi_i$. A *clause* is a disjunction of literals. A propositional formula is in *conjunctive normal form* (CNF) if it is a conjunction of clauses, and in *disjunctive normal form* (DNF) if it is a disjunction of conjunctions of literals (sometimes referred to as conjunctive clauses). Commonly used special cases are CNF or DNF formulas with at most three literals in each clause or conjunctive clause. Such formulas are said to be in 3-CNF or 3-DNF, respectively.

In the context of formal logic, the process of *reasoning* refers to the application of logical *inference*: given certain knowledge, derive further knowledge that can be logically inferred from it. Logical inference lies at the heart of formal arguments and thus plays a key role in establishing formal models that represent argumentation scenarios. We first illustrate logical inference in monotonic logic and then introduce non-monotonic inference as the core formalism for formal argumentation models. Our notation loosely follows that of Caminada and Amgoud (2007).

Let X be a set of propositional variables. A *knowledge base* over X is a set of literals over X, and represents the facts that are known to be true a priori. For now, we assume knowledge bases to be consistent. A strict *inference rule* allows to deductively infer a specific literal when a corresponding set of preconditional literals is known. We write a strict inference rule r as $r : \varphi_1, \ldots, \varphi_k \rightarrow \psi$, where r is the rule name, $\{\varphi_1, \ldots, \varphi_k\}$ is a consistent set of literals called the *body* of the rule, the arrow "$\rightarrow$" indicates strict logical inference, and ψ is the *head* of the rule. Rule r states that, when all literals $\varphi_1, \ldots, \varphi_k$ are known to hold, then ψ holds, too.

We call a pair consisting of a knowledge base and a set of inference rules over a common set of variables a *theory*. Reasoning in a theory is done by using known literals from the knowledge base to *activate* inference rules, where a rule is active if all of its body literals are known to hold. An active rule allows inferring the rule's head literal, which is then added to the knowledge, and which in turn may allow activating further rules. This iterative process of deductively deriving further knowledge by applying inference rules is formalized by the notion of *closure* in Definition 1.

Definition 1 (Closure)**.** Let X be a set of Boolean variables, K be a knowledge base over X and R be a set of inference rules over X. The *closure* $Cl_R(K)$ of K under R is the least set (with respect to set inclusion) that fulfills the following criteria:

1. $K \subseteq Cl_R(K)$,
2. for all $\varphi_1, \ldots, \varphi_k \rightarrow \psi \in R$ with $\varphi_1, \ldots, \varphi_k \in Cl_R(K)$, $\psi \in Cl_R(K)$ must hold.

The closure can be constructed using the following algorithm: Start with the known facts from the knowledge base and iteratively apply all inference rules that can be activated. Stop when no more new knowledge can be derived.

1. Let $K^0 = K$ and $i = 1$.
2. Let $R^{i-1} \subseteq R$ be the set of rules whose body literals are in K^{i-1}. Let $K^i = K^{i-1} \cup \mathit{Conc}(R^{i-1})$, were $\mathit{Conc}(R)$ denotes the union of all head literals of rules in the set R.
3. If $K^{i-1} \neq K^i$, set $i \leftarrow i + 1$ and repeat Steps 2 and 3.
4. The closure $\mathit{Cl}_R(K)$ is K^i.

In a deductive theory, a *proof* for a literal φ is a tree structure of inference rules, where φ is the root of the tree, literals in the knowledge base are the tree leaves, and each inference rule applied in the proof establishes the connection between its body literal nodes to its head literal node.

Example 2. Consider a set $R = \{r_1 : a \to b, r_2 : b, c \to e\}$ of inference rules and two knowledge bases $K_1 = \{c, \neg d\}$, $K_2 = \{a, c, \neg d\}$ over a set $\{a, b, c, d, e\}$ of variables.

- The closure $\mathit{Cl}_R(K_1)$ of K_1 is equal to K_1, since none of the two inference rules in R can be activated using this knowledge base.
- The closure $\mathit{Cl}_R(K_2)$ of K_2 is $\{a, b, c, \neg d, e\}$, since $a \in K_2$ allows to activate r_1 and infer b, so in the second iteration, both body literals b and c of r_2 are known, allowing to activate r_2 and also infer e.

In a theory consisting of K_2 and R, a proof for literal e is a tree with root e, leaves $a, c \in K_2$, and an inner node b. b and c are children of e and a is a child of b.

Deductive reasoning—i.e., deriving knowledge by applying strict inference rules using a knowledge base of priorly known facts—is *monotonic* in that it can only add new knowledge and never retract previous knowledge. This is formalized in Definition 3.

Definition 3 (Monotonicity)**.** A theory base is *monotonic* if for any set of inference rules R, any two consistent sets of facts K and K' with $K \subseteq K'$, and a literal φ, it holds that if $\varphi \in \mathit{Cl}_R(K)$, then $\varphi \in \mathit{Cl}_R(K')$.

Monotonic theories can be generalized by allowing inconsistent knowledge bases and *defeasible* inference rules. Opposed to strict rules, a defeasible inference rule does not guarantee its head literal to hold even if all body literals hold—there may be exceptions to its applicability. However, defeasible rules come with the significant advantage that they can represent non-deductive forms of inference like abduction, induction, a closed-world assumption, or commonsense reasoning, thus greatly increasing the scope of applications that can be represented by a theory. We write a defeasible inference rule r that defeasibly infers a head literal ψ from a set of body literals $\{\varphi_1, \ldots, \varphi_k\}$ as $r : \varphi_1, \ldots, \varphi_k \Rightarrow \psi$.

Applying defeasible inference rules to infer a literal creates not a proof, but only an *argument* for that literal. Like a proof, an argument can be represented by a tree structure with the head literal as its root, body literals from the knowledge base as leaves, and activated inference rules connecting inner nodes to their body literal nodes. But unlike a proof, an argument does not guarantee its head literal to hold, and different arguments in a theory do not need to be compatible and may be in conflict with each other. As a consequence, theories that incorporate defeasible inference rules are *non-monotonic*—knowledge that was inferred at some point of a reasoning process may be in conflict with new knowledge derived later, and may need to be retracted. In addition, while the closure provides a simple formalism to determine the set of all knowledge that can be believed in a monotonic theory, the closure of non-monotonic theories may be inconsistent. In order to identify consistent sets of literals that can be believed in a non-monotonic theory, more sophisticated methods are required.

A framework that allows creating arguments from defeasible theories is the ASPIC$^+$ framework introduced next in Section 2.2.1, while a framework that provides mechanisms to identify consistent sets of arguments is the model of abstract argumentation frameworks introduced thereafter in Section 2.2.2.

2.2 Formal Argumentation Models

In this section, we give an introduction to the formal argumentation models of the ASPIC$^+$ framework (Section 2.2.1), *abstract Argumentation Frameworks* (Section 2.2.2), and *Abstract Dialectical Frameworks* (Section 2.2.3), which are utilized in the technical part of this thesis. There exist a couple of other formal models of argumentation, for example, *Defeasible Logic Programming* (DeLP) proposed by García and Simari (2004) or *Assumption-based Argumentation Frameworks* (ABA) proposed by Bondarenko et al. (1993). For a broader introduction, see, e.g., the books by Rahwan and Simari (2009) or Baroni et al. (2018).

2.2.1 ASPIC$^+$

The ASPIC$^+$ framework was developed as a means to instantiate argument graphs from defeasible theories. ASPIC was first proposed in a technical report by Amgoud et al. (2004). Based on weaknesses of the ASPIC model (and other logic-based models of formal argumentation) identified by Caminada and Amgoud (2007), the extended framework ASPIC$^+$ was proposed by Prakken (2010) and later refined by Modgil and Prakken (2011) and Modgil and Prakken (2013), with a comprehensive introduction given by Modgil and Prakken (2014).

We formally define and illustrate ASPIC$^+$, following the model and notation from the latest revision by Modgil and Prakken (2013). In ASPIC$^+$, an argumentation system specifies the logical language within which the reasoning takes place and the rules of inference that can be applied to elements in this language. The language may be simply a set of literals closed under negation, or any more sophisticated logical language.

Definition 4 (Argumentation System)**.** An ASPIC$^+$ *argumentation system* is a quadruple $AS = (\mathcal{L}, ^-, \mathcal{R}, n)$, where:

- $\mathcal{L}$ is a logical language,
- $^- : \mathcal{L} \longrightarrow 2^{\mathcal{L}}$ is a contrariness function for literals, where $\varphi \in \mathcal{L}$ and $\psi \in \mathcal{L}$ are called contradictories of each other if $\varphi \in \bar{\psi}$ and $\psi \in \bar{\varphi}$, and where every element of $\mathcal{L}$ is assumed to have at least one contradictory,
- $\mathcal{R} = \mathcal{R}_s \uplus \mathcal{R}_d$ is a set of strict ($\mathcal{R}_s$) and defeasible ($\mathcal{R}_d$) inference rules, and
- n is a function $n : \mathcal{R}_d \longrightarrow \mathcal{L}$ that maps defeasible rules to their language representations and that allows rule identifiers to be used within the language.

An ASPIC$^+$ knowledge base specifies the prior knowledge in a reasoning process, and allows to distinguish between literals that are guaranteed to hold and those that may only be assumed to hold.

Definition 5 (Knowledge Base)**.** An ASPIC$^+$*knowledge base* in an argumentation system with language $\mathcal{L}$ is a set $\mathcal{K}$ with $\mathcal{K} \subseteq \mathcal{L}$ and $\mathcal{K} = \mathcal{K}_n \uplus \mathcal{K}_p$, where:

- $\mathcal{K}_n$ is the set of *axioms* which are guaranteed to be true, and
- $\mathcal{K}_p$ is the set of *ordinary premises* which may be assumed, but which can be defeated.

An ASPIC$^+$*argumentation theory AT* is the combination of an argumentation system AS and a knowledge base $\mathcal{K}$ over a common language $\mathcal{L}$. The purpose of an argumentation theory is to construct arguments which can be used to evaluate the dialectical status—accepted or rejected—of literals in the theory.

Definition 6 (Argument)**.** An *argument* A on the basis of an argumentation theory $AT = (AS, \mathcal{K})$ with $AS = (\mathcal{L}, ^-, \mathcal{R}, n)$ and $\mathcal{K} \subseteq \mathcal{L}$ consists of the following elements:

- $\mathit{Prem}(A) \subseteq \mathcal{L}$ – the set of premises,
- $\mathit{Conc}(A) \in \mathcal{L}$ – the conclusion,
- $\mathit{Sub}(A)$ – the set of subarguments of A,
- $\mathit{DefRules}(A) \subseteq \mathcal{R}_d$ – the set of defeasible inference rules applied in A, and
- $\mathit{TopRule}(A) \in (\mathcal{R}_s \cup \mathcal{R}_d)$ – the last inference rule applied in A.

Arguments in ASPIC$^+$ are built recursively from the knowledge base: A literal argument just states a single literal from the knowledge base without applying any inference rules, where for each $\varphi \in \mathcal{K}$, $A = \varphi$ is an argument with $\mathit{Prem}(\varphi) = \{\varphi\}$, $\mathit{Conc}(\varphi) = \varphi$, $\mathit{Sub}(\varphi) = \{\varphi\}$, $\mathit{DefRules}(\varphi) = \emptyset$, and $\mathit{TopRule}(\varphi) = \texttt{undefined}$. More advanced arguments apply a single inference rule and make use of subarguments which provide the body literals necessary to activate that inference rule.

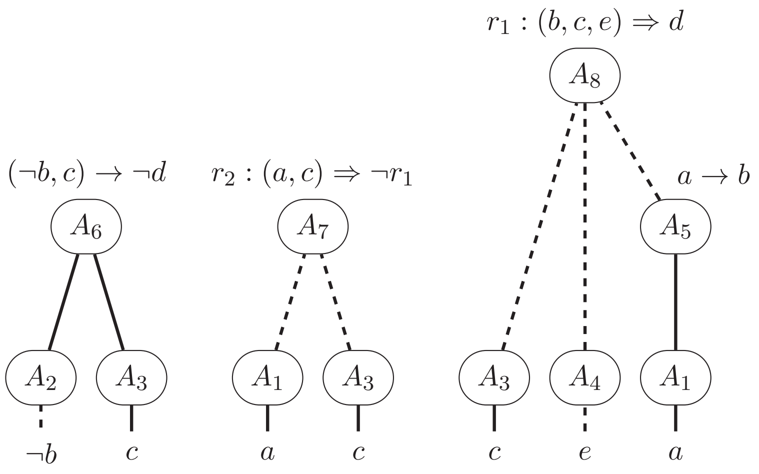

$r_1 : (b, c, e) \Rightarrow d$
A_8
$(\neg b, c) \rightarrow \neg d$
$r_2 : (a, c) \Rightarrow \neg r_1$
$a \rightarrow b$
A_6
A_7
A_5
A_2
A_3
A_1
A_3
A_3
A_4
A_1
$\neg b$
c
a
c
c
e
a

Figure 2: Tree visualizations of all ASPIC$^+$ arguments created in Example 8. Subarguments are displayed as children of an argument node, and literals are the tree leaves. All non-literal arguments are labeled with the inference rule that they apply. Dashed edges represent, depending on the context, either a defeasible inference rule or an ordinary premise. Arguments that appear as subarguments of several other arguments are displayed multiple times for improved readability.

Definition 7 (Strict and Defeasible Arguments)**.** If $A_1, \ldots, A_n$ are arguments and if there exists a strict rule $r \in \mathcal{R}_s$ with $r = Conc(A_1), \ldots, Conc(A_n) \rightarrow \psi$ (respectively, a defeasible rule $r \in \mathcal{R}_d$ with $r = Conc(A_1), \ldots, Conc(A_n) \Rightarrow \psi$), then A is an argument with:

- $Prem(A) = Prem(A_1) \cup \cdots \cup Prem(A_n)$,
- $Conc(A) = \psi$,
- $Sub(A) = Sub(A_1) \cup \cdots \cup Sub(A_n) \cup \{A\}$,
- $DefRules(A) = DefRules(A_1) \cup \cdots \cup DefRules(A_n) \cup (\{r\} \cap \mathcal{R}_d)$, and
- $TopRule(A) = r$.

ASPIC$^+$ allows to distinguish between certain and uncertain prior knowledge through the distinction between $\mathcal{K}_n$ and $\mathcal{K}_p$ in the knowledge base $\mathcal{K}$, and between strict and defeasible rules through the partition of $\mathcal{R}$ into $\mathcal{R}_s$ and $\mathcal{R}_d$. This allows to classify arguments based on the strength of the knowledge base literals and inference rules that they use. An ASPIC$^+$ argument A is called *strict* if $DefRules(A) = \emptyset$, and *defeasible* otherwise. A is called *firm* if $Prem(A) \subseteq \mathcal{K}_n$ (or equivalently, if $Prem(A) \cap \mathcal{K}_p = \emptyset$), and *plausible* otherwise. Every argument is either strict or defeasible and either firm or plausible, allowing a total of four individual argument type combinations. Strict-and-firm arguments represent deductive proofs of their conclusion, while all other argument types are non-deductive. Therefore, every strict-and-firm argument guarantees its conclusion, whereas a defeasible or plausible argument only provides a reason to believe its conclusion.

We illustrate ASPIC$^+$ argumentation theories and arguments in Example 8.

Example 8. Consider an ASPIC$^+$ argumentation theory given by the following argumentation system and knowledge base.

- $\mathcal{L} = \{a, b, c, d, e, r_1, r_2\} \cup \{\neg a, \neg b, \neg c, \neg d, \neg e, \neg r_1, \neg r_2\}$,
- $\bar{\varphi} = \{\neg\varphi\}$ for all $\varphi \in \mathcal{L}$,
- $\mathcal{R}_s = \{a \rightarrow b,\ (\neg b, c) \rightarrow \neg d\}$,
- $\mathcal{R}_d = \{(a, c) \Rightarrow \neg r_1,\ (b, c, e) \Rightarrow d\}$,
- $n((b, c, e) \Rightarrow d) = r_1,\ n((a, c) \Rightarrow \neg r_1) = r_2$.
- $\mathcal{K}_n = \{a, c\}$,
- $\mathcal{K}_p = \{\neg b, e\}$.

In this argumentation theory, four literal arguments can be created: $A_1 = a$ and $A_3 = c$ from $\mathcal{K}_n$, and $A_2 = \neg b$ and $A_4 = e$ from $\mathcal{K}_p$.

Using the strict inference rules in the argumentation system, two strict arguments can be created:

- From rule $a \rightarrow b$, create argument A_5 with subargument $A_1 = a$.
- From rule $(\neg b, c) \rightarrow \neg d$, create argument A_6 with subarguments $A_2 = \neg b$ and $A_3 = c$.

Using the defeasible inference rules in the argumentation system, two defeasible arguments can be created:

- From rule $(a, c) \Rightarrow \neg r_1$, create argument A_7 with subarguments $A_1 = a$ and $A_3 = c$.
- From rule $(b, c, e) \Rightarrow d$, create argument A_8 with subarguments $A_3 = c$, $A_4 = e$, and A_5 (with conclusion b).

All eight arguments are summarized in Table 1. Arguments A_7 and A_8 apply a defeasible rule and are therefore defeasible, while all other arguments do not apply defeasible rules and are strict. Arguments A_2 and A_6 use the ordinary premise $\neg b$, and arguments A_4 and A_8 use the ordinary premise e, and are therefore plausible, while the remaining arguments are firm. Figure 2 displays tree visualizations of all arguments that were created.

ASPIC$^+$ arguments can be in conflict with each other. ASPIC$^+$ distinguishes three different types of argument attacks. An undermining attack indicates that the conclusion of an argument is incompatible with a premise of another argument, a rebutting attack indicates that the conclusions of two arguments are mutually inconsistent, and an undercutting attack indicates that the conclusion of an argument is a direct attack against a defeasible inference rule applied in another argument.

Table 1: ASPIC$^+$ arguments created from the argumentation theory in Example 8

	Prem	**Conc**	**Sub**	**DefRules**	**TopRule**
$A_1 = a$	$\{a\}$	a	$\{a\}$	$\emptyset$	`undefined`
$A_2 = \neg b$	$\{\neg b\}$	$\neg b$	$\{\neg b\}$	$\emptyset$	`undefined`
$A_3 = c$	$\{c\}$	c	$\{c\}$	$\emptyset$	`undefined`
$A_4 = e$	$\{e\}$	e	$\{e\}$	$\emptyset$	`undefined`
A_5	$\{a\}$	b	$\{A_1, A_5\}$	$\emptyset$	$a \rightarrow b$
A_6	$\{\neg b, c\}$	$\neg d$	$\{A_2, A_3, A_6\}$	$\emptyset$	$(\neg b, c) \rightarrow \neg d$
A_7	$\{a, c\}$	$\neg r_1$	$\{A_1, A_3, A_7\}$	$\{(a, c) \Rightarrow \neg r_1\}$	$(a, c) \Rightarrow \neg r_1$
A_8	$\{a, c, e\}$	d	$\{A_1, A_3, A_4, A_5, A_8\}$	$\{(b, c, e) \Rightarrow d\}$	$(b, c, e) \Rightarrow d$

Definition 9 (Argument Attacks)**.** An ASPIC$^+$ argument A undermines argument B (on $\varphi \in \mathcal{L}$) if and only if:

- $\varphi \in \mathit{Prem}(B)$ is a premise of B,
- $\varphi \in \mathcal{K}_p$ is an ordinary premise of the knowledge base, and
- $\mathit{Conc}(A) \in \bar{\varphi}$.

An ASPIC$^+$ argument A rebuts argument B (on subargument B') if and only if:

- $B' \in \mathit{Sub}(B)$ is a subargument of B,
- $\mathit{TopRule}(B') \in \mathcal{R}_d$, and
- $\mathit{Conc}(B') = \varphi$ and $\mathit{Conc}(A) \in \bar{\varphi}$ for some $\varphi \in \mathcal{L}$.

An ASPIC$^+$ argument A undercuts argument B (on subargument B') if and only if:

- $B' \in \mathit{Sub}(B)$ is a subargument of B,
- $\mathit{TopRule}(B') \in \mathcal{R}_d$, and
- $\mathit{Conc}(A) \in \bar{n}(\mathit{TopRule}(B'))$.

Example 8 (continuing from p. 11)**.** In our example, we have the following argument attacks: A_5 undermines A_2 on $\neg b$, A_5 undermines A_6 on $\neg b$, A_6 rebuts A_8 on A_8, and A_7 undercuts A_8 on A_8.

Since all attacks target some defeasible element in the attacked argument (either an ordinary premise or a defeasible rule), each attack type can only target specific argument types: only plausible arguments can be undermined, while only defeasible arguments can be undercut or rebutted. On the other hand, each argument type can be the source of all kinds of attacks without limitation.

The ASPIC framework was extended to ASPIC$^+$ by Prakken (2010), who proposed to refine an argumentation theory by introducing preference orderings on its arguments that indicate relative strength. Preferences allow to distinguish between argument attacks—as defined here—and argument defeat, where an attack is a *defeat* if it is either an undercut or successful, and *successful* if the target argument is not preferred to the attacking argument. Preferences between arguments can be derived from a preference relation on knowledge base literals in $\mathcal{K}_p$ and a preference relation on defeasible inference rules in $\mathcal{R}_d$, which are part of an ASPIC$^+$ instance. For technical details, we refer the interested reader to Modgil and Prakken (2013).

ASPIC$^+$ is a general framework that allows the creation of custom argumentation systems, which in turn can be instantiated using specific data. It is important to avoid building a system that can produce counter-intuitive results—e.g., a set of arguments that do not attack each other, but whose conclusions are not consistent. In order to control the behavior of systems based on ASPIC$^+$, Caminada and Amgoud (2007) propose a set of *rationality postulates* that each impose constraints on the acceptable conclusions that are derived from a system.

Definition 10 (Rationality Postulates)**.** Let Args be a set of ASPIC$^+$ arguments.

- Args fulfills *subargument closure* if for any argument $A \in \mathit{Args}$, $\mathit{Sub}(A) \subseteq \mathit{Args}$.
- Args fulfills *closure under strict rules* if its set of conclusions is closed under strict closure, i.e., $\{\mathit{Conc}(A) \mid A \in \mathit{Args}\} = \mathit{Cl}_{\mathcal{R}_s}(\{\mathit{Conc}(A) \mid A \in \mathit{Args}\})$.
- Args fulfills *direct consistency* if $\{\mathit{Conc}(A) \mid A \in \mathit{Args}\}$ is consistent.
- Args fulfills *indirect consistency* if $\mathit{Cl}_{\mathcal{R}_s}(\{\mathit{Conc}(A) \mid A \in \mathit{Args}\})$ is consistent.

The postulates in Definition 10 can be generalized to be applicable also for other frameworks than ASPIC$^+$. An instantiated argumentation system satisfies one of the postulates if all sets of acceptable arguments produced by that system satisfy the postulate.

ASPIC$^+$ can be instantiated using the TOAST system by Snaith and Reed (2012), which was developed to represent defeasible theories following the ASPIC$^+$ format specifically. There are other related tools for reasoning tasks in structured knowledge bases that are not tailored to the ASPIC$^+$ framework, e.g., Argue-tuProlog by Bryant et al. (2006). A recent survey is given by Cerutti et al. (2017).

2.2.2 Abstract Argumentation Frameworks

Abstract argumentation frameworks were proposed by Dung (1995) as a simple model that provides formal methods to resolve conflicts in a graph structure consisting of arguments and the attack relation between them, and to determine which arguments can be considered acceptable. The model's abstract nature allows it to be used for a wide range of different scenarios.

We will now introduce all fundamental definitions of Dung's original model, while our notation loosely follows that of Dunne and Wooldridge (2009).

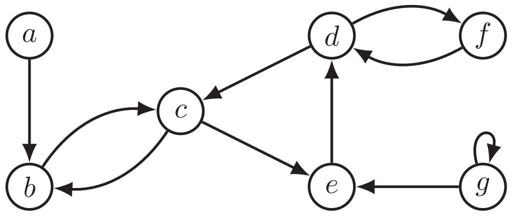


Figure 3: Graph representation of the argumentation framework in Example 12

Definition 11 (Argumentation Framework)**.** An *argumentation framework* (AF) is a pair $\langle \mathcal{A}, \mathcal{R} \rangle$ consisting of a finite set $\mathcal{A}$ of arguments and a binary attack relation $\mathcal{R} \subseteq (\mathcal{A} \times \mathcal{A})$ on $\mathcal{A}$. If $(a, b) \in \mathcal{R}$, this indicates that a attacks b.

We require the set $\mathcal{A}$ of arguments to be finite, since our work deals with complexity analysis of reasoning problems in argumentation frameworks on the one hand, and instantiation of real-world discussion data on the other hand. Finite instances are a precondition for complexity analysis, since the complexity is measured in relation to the input size. Further, infinite discussions do not occur in real applications, so this is a reasonable requirement. However, there are other works that consider argumentation frameworks where $\mathcal{A}$ may be infinite, e.g., Baumann and Spanring (2015).

Example 12. An AF $\langle \mathcal{A}, \mathcal{R} \rangle$ can be represented as a directed *graph* by identifying $\mathcal{A}$ as the set of nodes and $\mathcal{R}$ as the set of directed edges. Figure 3 displays an argumentation framework with arguments $\mathcal{A} = \{a, b, c, d, e, f, g\}$ and attacks $\mathcal{R} = \{(a, b), (b, c), (c, b), (c, e), (d, c), (d, f), (e, d), (f, d), (g, e), (g, g)\}$.

Although the identity of the arguments and the attacks between them are the only information that is available in an argumentation framework, it is enough to derive further insights about the relations between and dialectical status of the arguments. The first is the implicit notion of defense that is induced by the attacks. A set $Args \subseteq \mathcal{A}$ of arguments *defends* an argument $b \in \mathcal{A}$ if $Args$ attacks all attackers of b, i.e., if for each attacker $a \in \mathcal{A}$ with $(a, b) \in \mathcal{R}$ there exists a defender $d \in Args$ such that $(d, a) \in \mathcal{R}$. When $Args$ defends b, one may equivalently say that b *is acceptable with respect to* $Args$. The function that maps any set $Args$ of arguments in an argumentation framework AF to the set of all arguments defended by $Args$ is called the *characteristic function of* AF, and is formally defined as $F_{AF} : 2^{\mathcal{A}} \to 2^{\mathcal{A}}$ with $F_{AF}(Args) = \{a \in \mathcal{A} \,|\, a \text{ is defended by } Args \text{ in } AF\}$ for all $Args \subseteq \mathcal{A}$. The second notion derived from the attack relation is the property of *conflict-freeness*. A set $Args \subseteq \mathcal{A}$ of arguments is conflict-free if there are no attacks between members of $Args$, i.e., for all $a, b \in Args$ we have $(a, b) \notin \mathcal{R}$. The combined requirements of conflict-freeness and defense are captured by *admissibility*—a set $Args$ of arguments in an argumentation framework AF that is both conflict-free and that defends itself, i.e., $Args \subseteq F_{AF}(Args)$, is called admissible. Admissibility encompasses the intuition that a set of arguments cannot be acceptable if it contains arguments that are in conflict with each other, or if it cannot defend its members against attacks from other arguments. Admissibility is a precondition for most advanced criteria in argumentation frameworks and for all criteria considered in this thesis.

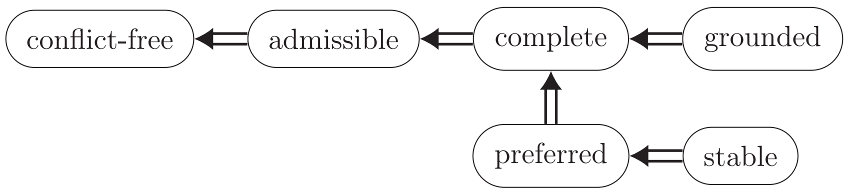


Figure 4: Implications between conflict-freeness, admissibility, and the four semantics from Definition 13, where for each pair of connected properties, all sets of arguments that satisfy the conditions of the parent property also satisfy the conditions of the child property.

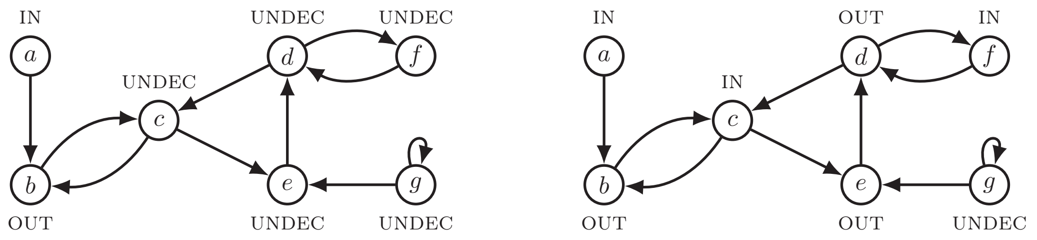
The acceptability status of a single argument in an argumentation framework is determined based on its membership in sets of arguments that are collectively acceptable (an equivalent approach based on argument labelings instead of sets of arguments was defined by Caminada (2006a)). A criterion that decides which sets of arguments in an argumentation framework can be considered acceptable is called a *semantics* of the argumentation framework. Many different semantics were proposed in the literature that serve different purposes, depending on the specific requirements of the application. We formally define the four semantics that were proposed by Dung (1995).

Definition 13 (Semantics)**.** Let $AF = \langle \mathcal{A}, \mathcal{R} \rangle$ be an argumentation framework and let $Args \subseteq \mathcal{A}$ be a conflict-free set of arguments.

- $Args$ is *complete* (CP) if it is a fixed point of the characteristic function of AF, i.e., if $Args = F_{AF}(Args)$.
- $Args$ is *grounded* (GR) if it is the unique least fixed point of the characteristic function of AF, i.e., if $Args = F^*_{AF}(\emptyset)$ (where F^*_{AF} denotes the infinite composition of F_{AF}).
- $Args$ is *preferred* (PR) if $Args$ is a set-maximal admissible set.
- $Args$ is *stable* (ST) if for every $b \in \mathcal{A} \setminus Args$ there is an $a \in Args$ with $(a, b) \in \mathcal{R}$.

Even though conflict-freeness and admissibility were introduced as basic preconditions and not as full-fledged semantics, there is no technical reason to differentiate between them and semantics, since they also provide a criterion to distinguish acceptable from unacceptable sets of arguments. We often use the shorthands CP, GR, PR, and ST to refer to the four semantics, CF for conflict-freeness, AD for admissibility, and $\mathbf{s}$ as a variable ranging over these. A set that satisfies the conditions of a semantics $\mathbf{s} \in \{\text{CF}, \text{AD}, \text{CP}, \text{GR}, \text{PR}, \text{ST}\}$ is called an $\mathbf{s}$ *extension* of AF.

The following implications hold between these properties, as proven by Dung (1995) (or, in some cases, as a direct consequence of their definition): Every stable set is preferred, every preferred set is complete, every complete set is admissible, and every admissible set is conflict-free. The unique grounded extension is complete, but need not be preferred, just as a preferred set need not be grounded. These implications are displayed in Figure 4. The grounded extension further coincides with the intersection of all complete extensions. It



(a) Grounded extension (b) Preferred extension

Figure 5: Visualization of both complete extensions in the AF of Example 12, where arguments are labeled IN if they are in the extension, labeled OUT if they are attacked by IN arguments, and labeled UNDEC otherwise.

can be algorithmically determined by applying the characteristic function F_{AF} repeatedly, starting from an empty set of arguments, until a fixed point is reached. In every argumentation framework, there is at least one preferred extension (which may be the empty set). However, there are argumentation frameworks that have no stable extension. Dung (1995) identifies restricted subclasses of argumentation frameworks, namely, *coherent* and *well-founded* argumentation frameworks, in which certain semantics coincide and that guarantee the existence of a stable extension. We do not utilize these notions here and therefore omit formal definitions.

Example 12 (continuing from p. 14)**.** We continue Example 12 and determine all extensions of the given argumentation framework. Its conflict-free sets of arguments are $\{a,c,f\}$, $\{a,d\}$, $\{a,f,e\}$, $\{b,d\}$, $\{b,f,e\}$, and all their subsets. Of these, only $\emptyset$, $\{a\}$, $\{f\}$, $\{a,f\}$, $\{c,f\}$, and $\{a,c,f\}$ defend all their members (and are therefore admissible), since argument a is unattacked, argument f defends itself against its only attacker d, and argument c defends itself against b and is defended by f against d.

To determine the complete extensions, we can check which of the admissible sets are fixed points of F_{AF}. We have $F_{AF}(\emptyset) = \{a\}$, $F_{AF}(\{a\}) = \{a\}$, $F_{AF}(\{f\}) = \{a,f\}$, $F_{AF}(\{a,f\}) = \{a,c,f\}$, $F_{AF}(\{c,f\}) = \{a,c,f\}$, and $F_{AF}(\{a,c,f\}) = \{a,c,f\}$. Thus, $\{a\}$ and $\{a,c,f\}$ are the only complete extensions of AF. Since $F_{AF}(\emptyset) = \{a\}$ and $F_{AF}(\{a\}) = \{a\}$, we also know that $\{a\}$ is the grounded extension of AF. The only set-maximal admissible set is $\{a,c,f\}$, which is the only preferred extension of AF. AF has no stable extension, because no conflict-free set has a chance to attack the self-attacking argument g.

Both complete extensions of this example are displayed in Figure 5 using the labeling representation by Caminada (2006a), where arguments are labeled IN if they are in the extension, labeled OUT if they are attacked by IN arguments, and labeled UNDEC otherwise.

The complete extensions of an argumentation framework are all admissible sets that are closed under defense, i.e., where no defended argument is deliberately excluded. Among the complete extensions, the grounded semantics represents a conservative approach—it only includes arguments that need to be accepted without doubt and suspends judgment for as many arguments as possible. On the other side of the spectrum, the stable semantics represents a decisive approach and requires judgment for all arguments: In a stable extension, every argument must either be in the extension (accepted) or attacked by the extension (rejected). Since this requirement is sometimes impossible to be satisfied, the preferred semantics serves as an intermediate, less demanding alternative between complete and stable semantics. A preferred extension locally maximizes judgment, but still allows judgment suspension if necessary, and thus guarantees the existence of a preferred extension. However, this comes at a computational price—as we will see in Section 2.3, most reasoning problems are harder to solve for the preferred than for the stable semantics.

In addition to Dung's original semantics used in this thesis, a range of further semantics has been proposed, which fill more roles not occupied by Dung's original semantics. Among them are the *stage* semantics by Verheij (1996), the *CF2* semantics by Baroni et al. (2005), the *semi-stable* semantics by Caminada (2006b), or the *ideal* semantics by Dung et al. (2006). A comprehensive overview is given in the survey by Baroni et al. (2011a).

The acceptability criteria that the various semantics provide for sets of arguments can be leveraged to derive acceptability criteria for individual arguments. Given an argumentation framework AF, a semantics $\mathbf{s}$ and a single argument $a \in \mathcal{A}$, a is *credulously acceptable* with respect to $\mathbf{s}$ if a is contained in at least one $\mathbf{s}$ extension of AF. Similarly, a is *skeptically acceptable* with respect to $\mathbf{s}$ if a is contained in all $\mathbf{s}$ extensions of AF. The notion of credulous and skeptical acceptance was initially proposed by Dunne and Bench-Capon (2002) for the preferred semantics, but is now established for all semantics.

Abstract AFs can be instantiated using the ASPIC$^+$ framework described in Section 2.2.1. As an alternative to ASPIC$^+$, a direct method to instantiate argumentation frameworks from defeasible theories was proposed by Wyner et al. (2015) and implemented by Straß (2014).

Several software tools exist that implement reasoning tasks for abstract argumentation frameworks. Among the most established are the answer-set programming (ASP) based solver ASPARTIX by Egly et al. (2010), the constraint satisfaction problem (CSP) based solver ConArg by Bistarelli and Santini (2011), and the SAT-based solvers CEGARTIX by Dvořák et al. (2012) and ArgSemSAT by Cerutti et al. (2014). Detailed surveys of different approaches to constructing AF solvers and of existing implementations are given by Charwat et al. (2015) and Cerutti et al. (2016).

2.2.3 Abstract Dialectical Frameworks

The formal model of abstract dialectical framework was developed by Brewka and Woltran (2010) and later revised by Brewka et al. (2013) as a powerful generalization of abstract argumentation frameworks. Like AFs, ADFs have the purpose of representing formal argumentations and of providing mechanisms to identify acceptable conclusions. The main difference between ADFs and AFs is that the conditions under which an individual element in the argumentation can be acceptable is explicitly specified in an ADF, while acceptability of all arguments in an AF uniformly depends on the attack relation and the semantics.

Definition 14 (Abstract Dialectical Framework)**.** An *abstract dialectical framework* (ADF) is a triple (S, L, C), where S is a set of *statements*, $L \subseteq S \times S$ is a set of directed *links* between statements, and $C = \{C_s\}_{s \in S}$ is a set of *acceptance conditions* for each statement.

Each link $(s_1, s_2) \in L$ indicates that the acceptance of statement s_2 depends on the acceptance of its parent statement s_1. For each statement s, its acceptance condition C_s defines how exactly its acceptance can be derived from the acceptance statuses of s's parents. C_s can be represented by either a Boolean function or a propositional formula over the acceptance statuses of s's parents. ADFs coincide with AFs in case the acceptance conditions are only attacks, i.e., if a statement is accepted when all its parents are not accepted.

ADFs borrow the notions of *interpretation* and *model* from logic programming to formalize how sets of accepted statements are to be determined. A (three-valued) interpretation $I : S \to \{\texttt{true}, \texttt{false}, \texttt{unknown}\}$ on a set S of ADF statements maps each statement s to either `true`, `false`, or `unknown`. An interpretation is called a model of the ADF if it conforms to the acceptance conditions, i.e., if for each statement s with $I(s) \neq \texttt{unknown}$, it holds that $I(s) = C_s[I]$, where $C_s[I]$ is the truth value that C_s evaluates to under I. Evaluation of formulas under three-valued interpretations is done following the standard evaluation algebra for three-valued logic. $\neg\texttt{unknown}$ is evaluated to `unknown`. $\varphi \wedge \psi$ is evaluated to `true` if both φ and ψ are `true`, to `false` if φ or ψ are `false`, and to `unknown` otherwise. $\varphi \vee \psi$ is evaluated to `true` if φ or ψ are `true`, to `false` if both φ and ψ are `false`, and to `unknown` otherwise. This evaluation clearly coincides with standard Boolean logic for two-valued interpretations, where no statement is mapped to `unknown`.

Example 15. Consider a set $S = \{a, b, c, d\}$ of statements and the following set $C = \{C_a, C_b, C_c, C_d\}$ of acceptance conditions:

$$\begin{aligned} C_a &= \neg C_c \vee (C_b \wedge \neg C_d) & \qquad C_b &= C_d \\ C_c &= \texttt{true} & C_d &= C_b \end{aligned}$$

The set of links $L = \{(b, a), (c, a), (d, a), (d, b), (b, d)\}$ is implicitly given through the acceptance conditions.

Consider an interpretation I_1 with $I_1(a) = \texttt{false}$, $I_1(b) = \texttt{true}$, $I_1(c) = \texttt{true}$, and $I_1(d) = \texttt{true}$. I_1 is a model of (S, L, C), because:

- $C_a[I_1] = \neg C_c[I_1] \vee (C_b[I_1] \wedge \neg C_d[I_1]) = \texttt{false} = I_1(a)$,
- $C_b[I_1] = C_d[I_1] = \texttt{true} = I_1(b)$,
- $C_c[I_1] = \texttt{true} = I_1(c)$, and
- $C_d[I_1] = C_b[I_1] = \texttt{true} = I_1(d)$.

Another model of (S, L, C) is I_2 with $I_2(a) = \texttt{false}$, $I_2(b) = \texttt{false}$, $I_2(c) = \texttt{true}$, and $I_2(d) = \texttt{false}$. Both I_1 and I_2 are even two-valued models. It is clear that no other two-valued interpretation is a model for this ADF.

The notion of models can be refined by different semantics—i.e., complete, grounded, stable, or preferred models—that are closely related to the corresponding semantics in abstract argumentation frameworks. For formal definitions, please refer to Brewka et al. (2013).

A method to instantiate abstract dialectical frameworks from defeasible theories was proposed by Straß (2015). Software tools to solve reasoning tasks in instantiated ADFs include DIAMOND by Ellmauthaler and Straß (2014), QADF by Diller et al. (2015), YADF by Brewka et al. (2017), and k++ADF by Linsbichler et al. (2018).

2.3 Computational Complexity

A majority of the technical results described in this thesis are classifications of decision problems with respect to their computational complexity. This section provides the required background in computational complexity theory. For a more general introduction to the topic, please refer to, e.g., Papadimitriou (1995) or Rothe (2005).

A *decision problem* specifies a set of admissible *instances*, which are assumed to be provided in some suitable encoding (typically binary), and a question to be decided for each instance, for which "yes" and "no" are the only possible answers. Many essential algorithmic challenges can be represented as decision problems. One of the most iconic decision problems is the satisfiability problem for Boolean formulas, abbreviated as SAT, which asks whether a given propositional formula is satisfiable, i.e., whether there is a truth assignment to the propositional variables in the formula for which the formula evaluates to `true`. We display decision problems in the following form, where the name of the problem is given at the top (possibly augmented by parameters and a shorthand in parentheses), followed by a specification of the admissible instances in the middle, and the problem question at the bottom.

Satisfiability (SAT)	
Given:	A formula φ on a set $X = \{x_1, \dots, x_n\}$ of propositional variables.
Question:	Does it hold that $\exists \tau_X \ : \varphi[\tau_X] = \texttt{true}$?

We call an instance I of a decision problem P where the answer is "yes" (respectively, "no") a "yes"-instance (respectively, a "no"-instance), and write $I \in P$ for "yes"-instances and $I \notin P$ for "no"-instances.

Example 16. Consider an instance $I = (\varphi, X)$ of SAT with the set $X = \{x_1, x_2\}$ of variables and formula $\varphi = x_1 \wedge (\neg x_1 \vee \neg x_2)$. It holds that $I \in$ SAT, because τ_X with $\tau_X(x_1) = \texttt{true}$ and $\tau_X(x_2) = \texttt{false}$ satisfies the condition $\varphi[\tau_X] = \texttt{true}$.

When we instead look at the instance $I' = (\varphi', X)$ with $\varphi' = x_1 \wedge (\neg x_1 \vee \neg x_2) \wedge x_2$, none of the four possible assignments on X satisfy φ', so $I' \notin$ SAT.

2.3.1 Complexity of Decision Problems in the Polynomial Hierarchy

The hardness of solving a given decision problem is measured by computational resources required by algorithms which solve that problem. An *algorithm* for a decision problem is a procedure that receives the encoding of an instance as input and computes the answer to the problem ("yes" or "no") as output. A key formalism to represent algorithms in theoretical computer science is the *Turing machine* invented by Turing (1937), which is a lightweight model of an abstract machine that, however, is still powerful enough to solve any algorithmically decidable decision problem. A formal definition of the Turing machine is not required to understand the following notions and is therefore omitted.

For a given algorithm (e.g., a Turing machine), we measure its *run time* by counting the number of elementary computing steps executed by the algorithm until the answer is produced. Each such step is required to run in constant time, i.e., a timespan independent of the individual features specific to the given instance. In particular, it must be independent of the size of the instance, which is the length of its encoding. We need to distinguish between *deterministic* algorithms and *non-deterministic* algorithms. For a deterministic algorithm, each execution of an elementary step maps the previous configuration of the algorithm to exactly one successor configuration. Non-deterministic algorithms allow multiple different successor configurations for a single configuration. In consequence, the execution of a deterministic algorithm is a linear sequence of configurations that lead to either an accepting configuration (indicating a "yes" answer for the input), a rejecting configuration (indicating a "no" answer), or that continues infinitely (which is equivalent to a "no" answer). On the other hand, in the execution of a non-deterministic algorithm, that sequence may split up into several execution branches at each step, producing an execution tree instead of a linear sequence. A non-deterministic algorithm for a decision problem outputs a "yes" answer if at least one of its branches reaches an accepting configuration, and a "no" answer otherwise.

The complexity of a decision problem is defined based on the general run time of algorithms for that problem. In order to formalize the general run time of an algorithm for all its possible instances, the run time is generalized as a function of the input size. When comparing the run time of an algorithm for different inputs, or that of different algorithms, we are not interested in run time factors that are independent of the input—since these are easily compensated by optimized implementations, faster computers, or a different encoding of the problem—but only those that change with varying size of the input. Also, the behavior of the algorithm for a few (i.e., finitely many) smaller instances may be ignored in favor of the behavior for the remaining (infinitely many) larger instances. That is, we want to capture only the *asymptotical* development of the run time for different inputs. Since the execution time of each elementary step of an algorithm is required to be independent of the input size, we can use the number of elementary steps executed by an algorithm as a measure of its asymptotical run time. Further, since we want to establish guarantees on the time required by algorithms, we focus on their *worst case* run time. For a fixed input size, the worst-case run time of an algorithm is the highest number of steps required by that algorithm to solve any instance of that size. The general asymptotical worst-case run time of an algorithm is the asymptotical development of its worst-case run time for all instances.

Example 17. For example, consider an algorithm that requires between $5n^2 + 10$ and $3 \cdot 2^{n-2} + 5$ elementary steps to find an answer for any instance, where n is the instance size. When we strip away constant elements in the run time, this leaves asymptotical run times between n^2 and 2^n . Since $n^2 < 2^n$ for all $n \geq 5$, the asymptotical worst-case run time of that algorithm is proportional to 2^n, and so belongs to the asymptotical growth class $\mathcal{O}(2^n)$.

Now, the computational complexity of a decision problem can be defined—based on the notion of the run time of algorithms—as the asymptotic worst-case run time of the "best" algorithm for that problem, i.e., the minimum asymptotic worst-case run time over all algorithms which solve that problem. Of course, it is infeasible to actually analyze all possible algorithms that solve a problem, so problem complexities are typically only bounded from above and below.

To obtain an upper bound on a problem's complexity, it is clearly sufficient to provide an algorithm with the respective run time. We divide the set of all decision problems into *classes* based on upper bounds to their complexity. We start with the classes that are based on Turing machines with a polynomial time bound.

Definition 18 (Complexity Classes)**.** The complexity classes P, NP, and coNP are defined as follows.

- P (for deterministic polynomial time) is the class of decision problems that can be solved by a deterministic Turing machine in a number of steps that is polynomial-bounded by the instance size.
- NP (for non-deterministic polynomial time) is the class of decision problems that can be solved by a non-deterministic Turing machine in a number of steps that is polynomial-bounded by the instance size.
- coNP (for complement NP) is the class of complements of NP problems.

NP can also be characterized as the class of decision problems for which a certificate claiming that the problem's answer for a given instance is "yes" can be verified in polynomial time. For coNP, an alternative characterization is the class of decision problems where a certificate for a "no" answer can be verified in polynomial time.

Example 19. SAT is in NP. An instance (φ, X) of SAT is a formula φ on a set X of variables. A certificate for $(\varphi, X) \in$ SAT is a satisfying assignment τ_X on X, i.e., an assignment where $\varphi[\tau_X] = \texttt{true}$. Given an instance (φ, X) and a certificate τ_X, it is clearly possible to evaluate $\varphi[\tau_X]$ in time that is polynomial-bounded by the size of the instance.

SAT's complementary problem is UNSAT, which takes the same type of instances and gives a "yes" answer for an instance if and only if SAT's answer is "no".

Unsatisfiability (UNSAT)	
Given:	A formula φ on a set $X = \{x_1, \dots, x_n\}$ of propositional variables.
Question:	Does it hold that $\forall \tau_X \ : \varphi[\tau_X] = \texttt{false}$?

UNSAT is in coNP. Again, an instance (φ, X) of UNSAT is a formula φ on a set X of variables, and a certificate for $(\varphi, X) \notin$ UNSAT is a satisfying assignment τ_X on X, which can be verified in polynomial time.

It is clear that P $\subseteq$ NP, since deterministic Turing machines are a special case of non-deterministic Turing machines. Further, the classes P and NP are assumed to be distinct. For example, while the current state of technology allows deterministic P algorithms to be directly implemented in real applications, non-deterministic NP algorithms can merely be emulated by deterministic algorithms that sequentially traverse their execution branches, instead of executing parallel branches simultaneously. Since the number of nodes in the execution tree may be exponentially higher than the length of each individual branch, this means that such an emulating deterministic algorithm may take exponentially more time to compute an answer than the hypothetical non-deterministic algorithm. Currently, no deterministic algorithm is known that can guarantee a polynomial run time for the hardest problems in NP, and it is assumed that no such algorithm can exist.

Various complexity classes exist beyond NP and coNP. We will introduce the classes of the polynomial hierarchy initially introduced by Meyer and Stockmeyer (1972) and Stockmeyer (1976). These require the notion of *oracles*. An algorithm that has access to an oracle of a complexity class $\mathcal{C}$ can use that oracle to get an answer to a problem in $\mathcal{C}$ in a single step. Oracles can be seen as black boxes, where the work needed to solve sub-problems that lie within $\mathcal{C}$ is not counted towards the run time of the algorithm invoking the oracle. The complexity class consisting of all problems that can be solved by a non-deterministic Turing machine which has access to an oracle for a class $\mathcal{C}$ is written as $\mathrm{NP}^{\mathcal{C}}$.

NP^{P} is equal to NP, since the polynomial work done by a P oracle could as well be performed by the NP algorithm itself. $\mathrm{NP}^{\mathrm{NP}}$, however, is a different class, and denoted as Σ_2^p. The class of complements of Σ_2^p problems is called Π_2^p, and can be characterized as $\mathrm{coNP}^{\mathrm{NP}}$. This hierarchy can be extended arbitrarily using the following recursion, where Σ_0^p is set to be equal to P as a base case.

Definition 20 (Polynomial Hierarchy)**.** The *polynomial hierarchy* is the union of all classes Δ_i^p, Σ_i^p, and Π_i^p, defined as

- $\Delta_{i+1}^p = \mathrm{P}^{\Sigma_i^p}$
- $\Sigma_{i+1}^p = \mathrm{NP}^{\Sigma_i^p}$
- $\Pi_{i+1}^p = \mathrm{coNP}^{\Sigma_i^p}$

with $\Delta_0^p = \Sigma_0^p = \Pi_0^p = \mathrm{P}$.

This definition also yields that $\Delta_1^p = \mathrm{P}^{\mathrm{P}} = \mathrm{P}$, $\Sigma_1^p = \mathrm{NP}^{\mathrm{P}} = \mathrm{NP}$, and $\Pi_1^p = \mathrm{coNP}^{\mathrm{P}} = \mathrm{coNP}$. It is known that each level i of the hierarchy is fully contained in each class of the next level $i+1$ of the hierarchy, i.e., $(\Delta_i^p \cup \Sigma_i^p \cup \Pi_i^p) \subseteq \Delta_{i+1}^p$, $(\Delta_i^p \cup \Sigma_i^p \cup \Pi_i^p) \subseteq \Sigma_{i+1}^p$, and $(\Delta_i^p \cup \Sigma_i^p \cup \Pi_i^p) \subseteq \Pi_{i+1}^p$. The relation between two consecutive levels i and $i+1$ of the polynomial hierarchy is similar to that between P and NP: Each problem in level $i+1$ can be solved deterministically by an exponential number of calls to an oracle for level i, but for the hardest problems in each level, no deterministic algorithm is known that guarantees at most a polynomial number of calls. Accordingly, it is assumed that the inclusions between the classes given above are strict and that each new level of the hierarchy contains inherently harder problems than the previous level. In the technical results of this thesis, upper complexity bounds on decision problems are provided in the form of membership within certain classes of the polynomial hierarchy.

The SAT problem, which is in $\mathrm{NP} = \Sigma_1^p$, can be extended to a sequence of quantified satisfiability problems which are typical representatives for the class Σ_i^p. The i-QUANTIFIED-SATISFIABILITY problem allows the propositional variables in an instance to be divided into i disjoint sets. In the problem question, the assignments on variables in the first set are existentially quantified, while the assignments on variables in the second set are universally quantified within the first quantifier. This continues, with alternating quantifiers, for all sets. It is easy to see that this problem coincides with SAT for $i = 1$.

i-QUANTIFIED-SATISFIABILITY (QSAT_i)	
Given:	i disjoint sets $X_1, \dots, X_i$ of n propositional variables and a formula φ on $\bigcup_{j=1}^{i} X_j$.
Question:	$\exists \tau_{X_1} : \forall \tau_{X_2} : \exists \tau_{X_3} : \dots \varphi[\tau_{X_1}, \dots, \tau_{X_i}] = \texttt{true}$?

Analogously to the complementary problem UNSAT for SAT, we define a complementary problem $\overline{\mathrm{QSAT}}_i$ for each QSAT_i. These problems are typical representatives for the classes Π_i^p.

i-QUANTIFIED-UNSATISFIABILITY ($\overline{\mathrm{QSAT}}_i$)	
Given:	i disjoint sets $X_1, \dots, X_i$ of n propositional variables and a formula φ on $\bigcup_{j=1}^{i} X_j$.
Question:	$\forall \tau_{X_1} : \exists \tau_{X_2} : \forall \tau_{X_3} : \dots \varphi[\tau_{X_1}, \dots, \tau_{X_i}] = \texttt{false}$?

Finding a lower bound on the complexity of a problem is much harder, because, in principle, this requires a proof that all algorithms for that problem have an asymptotic worst-case complexity at least as high as that lower bound. Instead, the notion of reducibility is commonly used to closely relate the lower complexity bounds of different problems. This way, known lower complexity bounds of a problem can be propagated to other problems.

Definition 21 ($\leq_{\mathrm{m}}^{\mathrm{p}}$-Reducibility)**.** A problem P_1 is *polynomial-time many-one reducible* ($\leq_{\mathrm{m}}^{\mathrm{p}}$-reducible) to another problem P_2—written as $P_1 \leq_{\mathrm{m}}^{\mathrm{p}} P_2$—if there exists a translation f from instances of P_1 to instances of P_2 that satisfies the following properties:

- an instance I_1 of P_1 is a "yes"-instance of P_1 if and only if $f(I_1)$ is a "yes"-instance of P_2 (formally, $I_1 \in P_1 \iff f(I_1) \in P_2$), and
- computing $f(I_1)$ requires time that is polynomial-bounded by the size of I_1.

When a problem P_1 is $\leq_{\mathrm{m}}^{\mathrm{p}}$-reducible to problem P_2, this means that instances of P_1 can be simulated by solving certain instances of P_2 instead, while incurring only a polynomial workload overhead. As a consequence, if a lower bound on the asymptotical worst-case complexity of P_1 is known, then P_2 inherits a lower bound that is different from that of P_1 by at most a polynomial factor (due to the polynomial work that may be done by the function f in the process of the reduction).

Further, it is easy to see that if $P_1 \leq_{\mathrm{m}}^{\mathrm{p}} P_2$ via translation f and $P_2 \leq_{\mathrm{m}}^{\mathrm{p}} P_3$ via translation f', then $P_1 \leq_{\mathrm{m}}^{\mathrm{p}} P_3$ via the composed translation $f' \circ f$. Since f is polynomial-bounded in the size of instances I_1 of P_1 and f' is polynomial-bounded in $f(I_1)$, the composition $f' \circ f$ is polynomial in I_1, too. Thus, the relation over decision problems induced by $\leq_{\mathrm{m}}^{\mathrm{p}}$-reducibility is transitive.

The complexity classes of the polynomial hierarchy are closed under $\leq_{\mathrm{m}}^{\mathrm{p}}$-reduction. This allows identifying the hardest problems in each class—these are the problems to which all other problems in the class can be reduced.

Definition 22 (Hardness, Completeness)**.** A problem P is *hard* for a complexity class $\mathcal{C}$ if for each problem $P_{\mathcal{C}} \in \mathcal{C}$, it holds that $P_{\mathcal{C}} \leq_{\mathrm{m}}^{\mathrm{p}} P$—i.e., any problem in $\mathcal{C}$ can be reduced to P. A problem P is *complete* for a complexity class $\mathcal{C}$ if P is hard for $\mathcal{C}$ and $P \in \mathcal{C}$.

Native proofs for the NP-hardness of SAT were independently published by Cook (1971) and Levin (1973). Since we also know that SAT $\in$ NP, it follows that SAT is NP-complete. Due to the transitive nature of hardness, this result allows proving NP-hardness for any problem by simply reducing SAT to it. Garey and Johnson (1979) compiled a large collection of problems that are complete for NP and other classes. Reducing from any of these classes is enough to prove hardness of a new decision problem for the corresponding class.

QSAT_i is complete for Σ_i^p and $\overline{\mathrm{QSAT}}_i$ is complete for Π_i^p. The hardness remains even if these problems are restricted to instances where the formula is in conjunctive normal form with at most three literals per clause (3-CNF).

In the technical results of this thesis, lower complexity bounds on decision problems are provided in the form of hardness for certain classes of the polynomial hierarchy. We use the following instantiations of QSAT_i and $\overline{\mathrm{QSAT}}_i$, limited to 3-CNF instances, which are complete for NP (3-SAT), coNP (3-UNSAT), Σ_2^p (Σ_2SAT), Π_2^p (Π_2SAT), and Σ_3^p (Σ_3SAT).

3-CNF QSAT_1 (3-SAT)

Given: A set X of propositional variables and a 3-CNF formula φ on X.
Question: $\exists \tau_X : \varphi[\tau_X] = \texttt{true}$?

3-CNF $\overline{\text{QSAT}}_1$ (3-UNSAT)

Given: A set X of propositional variables and a 3-CNF formula φ on X.
Question: $\forall \tau_X : \varphi[\tau_X] = \texttt{false}$?

3-CNF QSAT_2 (Σ_2SAT)

Given: Two disjoint sets X and Y of propositional variables and a 3-CNF formula φ on $X \cup Y$.
Question: $\exists \tau_X : \forall \tau_Y : \varphi[\tau_X, \tau_Y] = \texttt{false}$?

3-CNF $\overline{\text{QSAT}}_2$ (Π_2SAT)

Given: Two disjoint sets X and Y of propositional variables and a 3-CNF formula φ on $X \cup Y$.
Question: $\forall \tau_X : \exists \tau_Y : \varphi[\tau_X, \tau_Y] = \texttt{true}$?

3-CNF QSAT_3 (Σ_3SAT)

Given: Three disjoint sets X, Y, and Z of propositional variables and a 3-CNF formula φ on $X \cup Y \cup Z$.
Question: $\exists \tau_X : \forall \tau_Y : \exists \tau_Z : \varphi[\tau_X, \tau_Y, \tau_Z] = \texttt{true}$?

2.3.2 Complexity of Reasoning Problems in Abstract Argumentation

In this thesis, three existing decision problems for argumentation frameworks—each of them parameterized by an evaluation semantics—are extended to a more general model. These problems represent central reasoning tasks for argumentation frameworks. For a given semantics **s**, **s**-Verification requires an argumentation framework $\langle \mathcal{A}, \mathcal{R} \rangle$ and a subset $S \subseteq \mathcal{A}$ of the arguments as input and asks whether S is an extension of $\langle \mathcal{A}, \mathcal{R} \rangle$ with respect to semantics **s**.

s-Verification (**s**-Ver)

Given: An argumentation framework $\langle \mathcal{A}, \mathcal{R} \rangle$ and a subset $S \subseteq \mathcal{A}$.
Question: Is S an **s** extension?

Both ACCEPTANCE problems take a single argument $a \in \mathcal{A}$ as input instead of a set of arguments, and ask whether a is contained in at least one $\mathbf{s}$ extension of the given argumentation framework (for $\mathbf{s}$-CREDULOUS-ACCEPTANCE) or even in all $\mathbf{s}$ extensions of the given argumentation framework (for $\mathbf{s}$-SKEPTICAL-ACCEPTANCE).

$\mathbf{s}$-CREDULOUS-ACCEPTANCE ($\mathbf{s}$-CA)	
Given:	An argumentation framework $\langle \mathcal{A}, \mathcal{R} \rangle$ and an argument $a \in \mathcal{A}$.
Question:	Is there an $\mathbf{s}$ extension S of $\langle \mathcal{A}, \mathcal{R} \rangle$ with $a \in S$?

$\mathbf{s}$-SKEPTICAL-ACCEPTANCE ($\mathbf{s}$-SA)	
Given:	An argumentation framework $\langle \mathcal{A}, \mathcal{R} \rangle$ and an argument $a \in \mathcal{A}$.
Question:	For all $\mathbf{s}$ extensions S of $\langle \mathcal{A}, \mathcal{R} \rangle$, does $a \in S$ hold?

There are a few special cases for these problems. Firstly, credulous and skeptical acceptance for the grounded semantics coincide, since there is always only one grounded extension. Further, since the grounded extension of an argumentation framework is the intersection of all its complete sets, an argument is in all complete extensions of an argumentation framework if and only if it is in its grounded extension, so the skeptical acceptance problem for the complete semantics coincides with that for the grounded semantics. Similarly, the union of all admissible sets, of all complete extensions, and of all preferred extensions of an argumentation framework are the same. Therefore, an argument is in at least one extension for one of these semantics if and only if it is in at least one extension for any of the others, so the credulous acceptance problems for the admissible, complete, and preferred semantics coincide. Finally, since there are argumentation frameworks that have no stable extension, ST-SKEPTICAL-ACCEPTANCE is not uniquely defined. There exist two versions of this problem—the first applies the rule that a universal quantifier over an empty set evaluates to `true`, so all ST-SA instances with an AF that has no stable extension are treated as “yes”-instances. An alternative version incorporates an exception for such instances and treats them as “no”-instances. In this thesis, we only cover the first version.

The $\mathbf{s}$-VER, $\mathbf{s}$-CA, and $\mathbf{s}$-SA problems have already been fully classified with respect to their computational complexity. Table 2 summarizes all results for the conflict-free (CF), admissible (AD), stable (ST), complete (CP), grounded (GR), and preferred (PR) semantics, and gives credit to the respective authors.

We give a short explanation for the straightforward cases from Table 2. A set $S \subseteq \mathcal{A}$ is conflict-free if and only if no attack exists among members of S, which can be checked in linear time. A single argument $a \in \mathcal{A}$ is contained in at least one conflict-free extension if and only if it does not attack itself. The empty set is always conflict-free and admissible, so the answer to CF-SA and AD-SA is always “no” regardless of the instance, making these problems trivial. The unique grounded extension can be determined by the polynomial-time fixed-point algorithm sketched in Section 2.2.2. Therefore, verification of a set of arguments for the grounded semantics can be done by comparing the given set to the grounded extension,

Table 2: Overview of the computational complexity of **s**-Verification, **s**-Credulous-Acceptance and **s**-Skeptical-Acceptance, where results marked with ♠ are by Dung (1995), with ♣ by Dimopoulos and Torres (1996), and with ▲ by Dunne and Bench-Capon (2002). Results without a reference are straightforward and do not require a formal proof.

s	**s**-Ver	**s**-CA	**s**-SA
cf	$\in$ P	$\in$ P	trivial
ad	$\in$ P ♠	NP-c. ♣	trivial
st	$\in$ P ♠	NP-c. ♣	coNP-c. ♣
cp	$\in$ P ♠	NP-c. ♣	$\in$ P
gr	$\in$ P	$\in$ P	$\in$ P
pr	coNP-c. ♣	NP-c. ♣	Π_2^p-c. ▲

and both acceptance problems for the grounded semantics (and thus also the cp-SA = gr-SA problem) collapse into checking membership of the given argument in the grounded extension.

The complexity results indicate that reasoning with the conflict-free or grounded semantics can always be done efficiently. Further, verifying whether a given set of arguments is an extension is possible in polynomial time for all but the preferred semantics. All other problems—except when they are trivial or coincide with one of the previous problems—are hard for some class in the first or second level of the polynomial hierarchy, indicating that these cases may be hard to solve for large argumentations in applications.

Chapter 3

Verification in Incomplete Argumentation Frameworks

This chapter summarizes our work on the complexity of generalizations of the Verification decision problem for incomplete argumentation frameworks published in our paper [Baumeister et al. (2018d)]:

> Baumeister, D., Neugebauer, D., Rothe, J., and Schadrack, H. (2018d). Verification in incomplete argumentation frameworks. *Artificial Intelligence*, 264:1–26.

This paper summarizes and extends results from our previous papers published at ADT'15 in [Baumeister et al. (2015b,c)], at COMSOC'16 in [Baumeister et al. (2016)], at AAAI'18 in [Baumeister et al. (2018b)], and at COMSOC'18 in [Baumeister et al. (2018c)].

3.1 Summary

Unquantified uncertainty in instances of abstract argumentation frameworks was first proposed by Coste-Marquis et al. (2007) and Cayrol et al. (2007) in the form of *partial argumentation frameworks*. In a partial argumentation framework, a subset of the attacks is distinguished, where each of these attacks individually may or may not exist. In this paper, we complement the notion of uncertain attacks by proposing a corresponding notion of uncertain arguments which may or may not exist in a given argumentation framework. In total, these two notions allow to represent either purely *attack-incomplete argumentation frameworks* (which coincide with partial argumentation frameworks), purely *argument-incomplete argumentation frameworks*, as well as a general model of *incomplete argumentation frameworks* where there may be uncertainty about both attacks and arguments. All of these models coincide with standard argumentation frameworks in case the set of uncertain elements is empty, so they are true generalizations of argumentation frameworks.

The technical contribution of this paper is the proposal of two generalizations of the established Verification problem, namely the **s**-Possible-Verification and **s**-Necessary-Verification problems, and a full analysis of the computational complexity of both problems with respect to all semantics introduced by Dung (1995). Although the possible and necessary problem variants are potentially harder than **s**-Verification, we were able to find non-trivial polynomial-time algorithms that solve **s**-Necessary-Verification and the purely attack-incomplete version of **s**-Possible-Verification, using the admissible, complete, grounded, or stable semantics. This indicates that it may often be feasible to perform verification tasks even if the state of the argumentation is not fully known.

3.2 Personal Contribution

The model of incomplete argumentation frameworks and the "possible" and "necessary" generalizations of the **s**-Verification problem were developed jointly with my co-authors. Also, all straightforward results (Theorems 26 and 28, and all Corollaries) are joint work with my co-authors. Writing of all non-technical text in the paper was done in equal parts by all authors. All further technical results, except Theorems 43 and 44, are my contribution, including all critical completions—namely, optimistic, pessimistic, fixed, unfixed, and ungrounded completion—and all derived results.

Chapter 4

Credulous and Skeptical Acceptance in Incomplete Argumentation Frameworks

This chapter summarizes our work on the complexity of generalizations of the s-Credulous-Acceptance and s-Skeptical-Acceptance decision problems for incomplete argumentation frameworks published in our paper [Baumeister et al. (2018a)]:

> Baumeister, D., Neugebauer, D., and Rothe, J. (2018a). Credulous and skeptical acceptance in incomplete argumentation frameworks. In *Proceedings of the 7th International Conference on Computational Models of Argument*, volume 305 of *Frontiers in Artificial Intelligence and Applications*, pages 181–192. IOS Press.

4.1 Summary

Similar to the generalized s-Possible-Verification and s-Necessary-Verification variants for incomplete argumentation frameworks of the established s-Verification problem for argumentation frameworks presented in Chapter 3, this paper proposes a generalization of both the s-Credulous-Acceptance and s-Skeptical-Acceptance problems by considering a *possible* and a *necessary* variant of both problems. Again, the computational complexity of the resulting problems s-Possible-Credulous-Acceptance, s-Necessary-Credulous-Acceptance, s-Possible-Skeptical-Acceptance, and s-Necessary-Skeptical-Acceptance is potentially higher than the complexity of the respective base problem for the same semantics.

The technical contribution of this paper is a full characterization of the complexity of the four problem variants for all six original semantics, using a generic reduction from different variants of the quantified satisfiability problem. As opposed to our results for s-Necessary-Verification and for the purely attack-incomplete version of s-Possible-Verification, where we were able to find non-trivial polynomial-time algorithms for all but the preferred semantics, in this paper all results establish either an increase in complexity or some trivial collapse of complexity due to, e.g., a quantifier representation of a problem with two consecutive universal quantifiers. In particular, for the acceptance problem variants, the results do not change if we restrict instances to purely attack-incomplete or purely argument-incomplete argumentation frameworks. This indicates that possible and necessary variants of acceptance problems in incomplete argumentation frameworks are substantially harder than the respective variants of the verification problem.

4.2 Personal Contribution

Writing was done jointly with my co-authors. All technical results are my contribution.

Chapter 5

Generating Defeasible Theories from Real-World Discussions using D-BAS

This chapter summarizes our report on the DABASCO software tool for instantiating formal argumentation data from D-BAS discussions given in our paper [Neugebauer (2019)] submitted to the “Argument & Computation” journal:

> Neugebauer, D. (2019). DABASCO: Generating defeasible theories from real-world discussions using D-BAS. *Argument & Computation.* Submitted.

Preliminary versions of this paper were published at AIAIAI’17 in [Neugebauer (2017)] and at COMMA’18 in [Neugebauer (2018)].

5.1 Summary

This work connects the discussion software D-BAS developed by Krauthoff et al. (2016, 2018) with the rich evaluation machinery that exists for the formal models of abstract argumentation frameworks, abstract dialectical frameworks, and the ASPIC$^+$ framework. The state of a D-BAS discussion at a given point in time is represented by a set of statements, a set of arguments, and a user opinion for each participant of the discussion. Each statement is a short natural language text, while each argument is a logical connective stating that believing the argument’s premise statements is a reason to believe the argument’s conclusion. Statements and arguments can be created by operators of a D-BAS platform as a starting point of a discussion, as well as by users during a discussion. A user opinion specifies which statements are accepted and which are rejected by that user in the discussion.

In our work, we first formalize the state of a D-BAS discussion as a defeasible theory. We provide the option of incorporating either no user opinion, which yields an objective representation of the discussion, or a single user’s opinion, which produces a subjective representation from that user’s point of view. If a user opinion is used, it can be encoded using either strict or defeasible inference rules, which affects the strength of the opinion in the theory. For the defeasible theories thus created, we verify constraints introduced by Wyner et al.

(2015). Next, we adapt established translations that create instances of formal argumentation models from defeasible theories: the translation to AFs by Wyner et al. (2015) and the translation to ADFs by Straß (2015). We show that Wyner et al.'s notion of *well-formed preferred extensions* is flawed and propose the notion of *strongly well-formed preferred extensions* as a solution. Our adaptation of Wyner et al.'s translation guarantees all preferred extensions in the resulting AF to be strongly well-formed. In addition, we present a direct translation of our defeasible theories to ASPIC$^+$ instances. For all our translations, we prove that the rationality postulates of Caminada and Amgoud (2007) are satisfied.

All translations are implemented in the software module DABASCO, which is available on GitHub at `https://github.com/hhucn/dabasco` as open-source software. DABASCO is a Python program that communicates with D-BAS via a REST interface and produces output that is compatible with existing solver software, including the TOAST online service by Snaith and Reed (2012) for ASPIC$^+$; ASPARTIX by Egly et al. (2010) and ConArg by Bistarelli and Santini (2011) for abstract AFs; and DIAMOND by Ellmauthaler and Straß (2014), YADF by Brewka et al. (2017), and k++ADF by Linsbichler et al. (2018) for ADFs.

5.2 Personal Contribution

All writing, implementation, and technical results are my contribution.

Chapter 6

Conclusions and Future Work

In this chapter, we give a summary of the contributions of this thesis and point out possible directions to continue this work. The objective of the research presented in this thesis was to narrow the gap that exists between formal models of argumentation in the computer science and AI research community on the one hand, and implemented systems for argumentation among humans on the other hand. The results establish progress on both sides of that gap.

6.1 Results

On the theoretical side, a powerful generalization of abstract argumentation frameworks was proposed which can represent uncertain information about the state of an argumentation and thus allows a wider range of natural situations to be captured by the model. Generalizations of the Verification, Credulous-Acceptance, and Skeptical-Acceptance decision problems were defined and their computational complexity was fully determined. A summary of our complexity results for all problems is given in Table 3. Our results have been published in peer-reviewed journals or conference proceedings in the form of the following articles: [Baumeister et al. (2015b)], [Baumeister et al. (2016)], [Baumeister et al. (2018b)], [Baumeister et al. (2018c)], [Baumeister et al. (2018d)], and [Baumeister et al. (2018a)].

Our complexity classifications help with deciding which of the generalized problems can realistically be solved in practice. While the Necessary-Verification problem for the different semantics has no higher asymptotical complexity than Verification, and the Possible-Verification problem is harder than Verification only for a few semantics, the possible and necessary variants of Credulous-Acceptance and Skeptical-Acceptance are harder than their respective base problem in all non-trivial cases, reaching hardness for the second or even the third level of the polynomial hierarchy. The results indicate that, for most semantics, solving the generalized possible or necessary problem variant is significantly harder for the acceptance problems than for the verification problem. In applications, it may therefore be feasible to solve generalized verification problems in a wider range of situations with incomplete information.

Table 3: Summarized complexity results for six semantics (first column) of the possible and necessary variants of VER (second to fourth column), CA (fifth and sixth column), and SA (seventh and eighth column). Results marked with ★ are partially due to Coste-Marquis et al. (2007). $\mathcal{C}$-c. is a shorthand for completeness in a complexity class $\mathcal{C}$.

s	s-PV		s-NV	s-PCA	s-NCA	s-PSA	s-NSA
	ATTINC	ARGINC/INC					
CF	$\in$ P★	$\in$ P	$\in$ P★	$\in$ P	$\in$ P	trivial	trivial
AD	$\in$ P	NP-c.	$\in$ P★	NP-c.	Π_2^p-c.	trivial	trivial
ST	$\in$ P	NP-c.	$\in$ P	NP-c.	Π_2^p-c.	Σ_2^p-c.	coNP-c.
CP	$\in$ P	NP-c.	$\in$ P	NP-c.	Π_2^p-c.	NP-c.	coNP-c.
GR	$\in$ P	NP-c.	$\in$ P	NP-c.	coNP-c.	NP-c.	coNP-c.
PR	Σ_2^p-c.	Σ_2^p-c.	coNP-c.	NP-c.	Π_2^p-c.	Σ_3^p-c.	Π_2^p-c.

On the practical side, we developed translation procedures that allow to generate AF, ADF, or ASPIC$^+$ representations of discussions created by users of the D-BAS platform by Krauthoff et al. (2018). The translations are shown to satisfy established quality criteria. All translations were implemented in the software tool DABASCO, which is available as free open-source software on GitHub at `https://github.com/hhucn/dabasco`. Our reports on DABASCO have been published in [Neugebauer (2017)] and [Neugebauer (2018)] and submitted for publication in [Neugebauer (2019)].

DABASCO connects D-BAS to solver software for reasoning problems in the different argumentation models, and thus establishes a full argument evaluation pipeline. This pipeline allows both the automatic analysis of D-BAS discussions, as well as the use of real-world discussions as benchmark data for solvers.

6.2 Future Work Directions

An immediate next step in the study on uncertainty in argumentation frameworks is the development of a data format that adequately represents instances of incomplete AFs, and the implementation of solver software for reasoning tasks on such instances, in order to transfer our theoretical results in this area to the practice.

On the theoretical side, our research on incomplete argumentation frameworks can be continued by analyzing further decision problems and by considering further semantics. Existing decision problems that could be generalized for incomplete argumentation frameworks include the ST-EXISTENCE problem, which examines the existence of a stable extension in an AF; the s-NON-EMPTINESS problem, which is a refinement of the s-EXISTENCE problem that considers only non-empty extensions (and which therefore is non-trivial not only for the stable semantics); and the Coherent problem defined by Dunne and Bench-Capon (2002), which determines whether a given AF satisfies the *coherence* property proposed by Dung (1995). Possible other semantics include the stage semantics by Verheij (1996), the CF2 semantics by Baroni et al. (2005) the semi-stable semantics by Caminada (2006b), or the ideal semantics by Dung et al. (2006).

From a more conceptual point of view, another interesting advancement of research on incomplete argumentation frameworks could be to allow interdependencies between uncertain elements of an argumentation—currently, the uncertainty of the possible arguments and attacks is assumed to be unrelated, meaning that every completion that is syntactically possible is assumed to be actually viable. However, there are several applications where the uncertainty of individual elements in an argumentation may be related. For example, different arguments may share common premises, or apply the same inference rule. If the existence of any of these elements in the underlying data is uncertain, then this uncertainty translates to a correlated uncertainty of all arguments that incorporate these elements. A model of incomplete argumentation frameworks with interdependencies would allow to represent such situations and further increase the scope of the model.

The software tool dabasco can be improved by allowing additional options to customize the encoding of a d-bas discussion, or by extending the set of argumentation models that a discussion can be translated to, such as assumption-based argumentation frameworks by Bondarenko et al. (1993), argumentation frameworks with recursive attacks by Baroni et al. (2011b), probabilistic argumentation frameworks by Li et al. (2011), or our model of incomplete argumentation frameworks.

Further, the evaluation pipeline consisting of d-bas, dabasco, and solver tools could be integrated into a feedback loop for d-bas by developing new d-bas features that directly utilize the results of the pipeline. Such features could provide helpful information for participants of a discussion, as well as insights for the operators of a d-bas platform.

Bibliography

Amgoud, L., Caminada, M., Cayrol, C., Lagasquie-Schiex, M., and Prakken, H. (2004). Towards a consensual formal model: inference part. Technical report, ASPIC project.

Baldwin, P. and Price, D. (2008). Debategraph. `http://debategraph.org` [Accessed: 2018-10-31].

Baroni, P., Caminada, M., and Giacomin, M. (2011a). An introduction to argumentation semantics. *The Knowledge Engineering Review*, 26(4):365–410.

Baroni, P., Cerutti, F., Giacomin, M., and Guida, G. (2011b). AFRA: Argumentation framework with recursive attacks. *International Journal of Approximate Reasoning*, 52(1):19–37.

Baroni, P., Gabbay, D., and Giacomin, M., editors (2018). *Handbook of Formal Argumentation*. College Publications.

Baroni, P., Giacomin, M., and Guida, G. (2005). SCC-recursiveness: A general schema for argumentation semantics. *Artificial Intelligence*, 168(1-2):162–210.

Baumann, R. and Brewka, G. (2010). Expanding argumentation frameworks: Enforcing and monotonicity results. In *Proceedings of the 3rd International Conference on Computational Models of Argument*, volume 216 of *Frontiers in Artificial Intelligence and Applications*, pages 75–86. IOS Press.

Baumann, R. and Spanring, C. (2015). Infinite argumentation frameworks. In *Advances in Knowledge Representation, Logic Programming, and Abstract Argumentation*, pages 281–295. Springer *Lecture Notes in Computer Science #9060*.

Baumeister, D., Erdélyi, G., Erdélyi, O., and Rothe, J. (2015a). Complexity of manipulation and bribery in judgment aggregation for uniform premise-based quota rules. *Mathematical Social Sciences*, 76:19–30.

Baumeister, D., Neugebauer, D., and Rothe, J. (2015b). Verification in attack-incomplete argumentation frameworks. In *Proceedings of the 4th International Conference on Algorithmic Decision Theory*, pages 341–358. Springer-Verlag *Lecture Notes in Artificial Intelligence #9346*.

Baumeister, D., Neugebauer, D., and Rothe, J. (2018a). Credulous and skeptical acceptance in incomplete argumentation frameworks. In *Proceedings of the 7th International Conference on Computational Models of Argument*, volume 305 of *Frontiers in Artificial Intelligence and Applications*, pages 181–192. IOS Press.

Baumeister, D., Neugebauer, D., Rothe, J., and Schadrack, H. (2016). Verification in incomplete argumentation frameworks. In *6th International Workshop on Computational Social Choice*.

Baumeister, D., Neugebauer, D., Rothe, J., and Schadrack, H. (2018b). Complexity of verification in incomplete argumentation frameworks. In *Proceedings of the 32nd AAAI Conference on Artificial Intelligence*, pages 1753–1760. AAAI Press.

Baumeister, D., Neugebauer, D., Rothe, J., and Schadrack, H. (2018c). Complexity of verification in incomplete argumentation frameworks. In *7th International Workshop on Computational Social Choice*.

Baumeister, D., Neugebauer, D., Rothe, J., and Schadrack, H. (2018d). Verification in incomplete argumentation frameworks. *Artificial Intelligence*, 264:1–26.

Baumeister, D., Rothe, J., and Schadrack, H. (2015c). Verification in argument-incomplete argumentation frameworks. In *Proceedings of the 4th International Conference on Algorithmic Decision Theory*, pages 359–376. Springer-Verlag *Lecture Notes in Artificial Intelligence #9346*.

Besnard, P. and Hunter, A. (2008). *Elements of Argumentation*. MIT Press.

Bex, F., Lawrence, J., Snaith, M., and Reed, C. (2013). Implementing the argument web. *Communications of the ACM*, 56(10):66–73.

Bex, F., Snaith, M., Lawrence, J., and Reed, C. (2014). ArguBlogging: An application for the argument web. *Web Semantics: Science, Services and Agents on the World Wide Web*, 25:9–15.

Bistarelli, S. and Santini, F. (2011). ConArg: A constraint-based computational framework for argumentation systems. In *Proceedings of the 23rd IEEE International Conference on Tools with Artificial Intelligence*, pages 605–612. IEEE.

Boella, G., Kaci, S., and van der Torre, L. (2009). Dynamics in argumentation with single extensions: Abstraction principles and the grounded extension. In *Proceedings of the 10th European Conference on Symbolic and Quantitative Approaches to Reasoning and Uncertainty*, pages 107–118. Springer-Verlag *Lecture Notes in Artificial Intelligence #5590*.

Bondarenko, A., Toni, F., and Kowalski, R. (1993). An assumption-based framework for non-monotonic reasoning. In *Proceedings of the 2nd International Workshop on Logic Programming and Non-monotonic Reasoning*, volume 93, pages 171–189.

Bouveret, S., Endriss, U., and Lang, J. (2010). Fair division under ordinal preferences: Computing envy-free allocations of indivisible goods. In *Proceedings of the 19th European Conference on Artificial Intelligence*, pages 387–392. IOS Press.

Brewka, G., Diller, M., Heissenberger, G., Linsbichler, T., and Woltran, S. (2017). Solving advanced argumentation problems with answer-set programming. In *Proceedings of the 31th AAAI Conference on Artificial Intelligence*, pages 1077–1083.

Brewka, G., Ellmauthaler, S., Straß, H., Wallner, J., and Woltran, S. (2013). Abstract dialectical frameworks revisited. In *Proceedings of the 23rd International Joint Conference on Artificial Intelligence*, pages 803–809. AAAI Press/IJCAI.

Brewka, G., Polberg, S., and Woltran, S. (2014). Generalizations of Dung frameworks and their role in formal argumentation. *IEEE Intelligent Systems*, 29(1):30–38.

Brewka, G. and Woltran, S. (2010). Abstract dialectical frameworks. In *Proceedings of the 12th International Conference on Principles of Knowledge Representation and Reasoning*, pages 102–111. AAAI Press.

Bryant, D., Krause, P., and Vreeswijk, G. (2006). Argue tuProlog: A lightweight argumentation engine for agent applications. *Proceedings of the 1st International Conference on Computational Models of Argument*, 144:27–32.

Caminada, M. (2006a). On the issue of reinstatement in argumentation. In *Proceedings of the 10th European Conference on Logics in Artificial Intelligence*, pages 111–123. Springer-Verlag *Lecture Notes in Artificial Intelligence #4160*.

Caminada, M. (2006b). Semi-stable semantics. In *Proceedings of the 1st International Conference on Computational Models of Argument*, volume 144 of *Frontiers in Artificial Intelligence and Applications*, pages 121–130. IOS Press.

Caminada, M. and Amgoud, L. (2007). On the evaluation of argumentation formalisms. *Artificial Intelligence*, 171(5):286–310.

Carstens, L., Fan, X., Gao, Y., and Toni, F. (2015). An overview of argumentation frameworks for decision support. In *International Workshop on Graph Structures for Knowledge Representation and Reasoning*, pages 32–49. Springer *Lecture Notes in Computer Science #9501*.

Cayrol, C., de Saint-Cyr, F., and Lagasquie-Schiex, M. (2010). Change in abstract argumentation frameworks: Adding an argument. *Journal of Artificial Intelligence Research*, 38:49–84.

Cayrol, C., Devred, C., and Lagasquie-Schiex, M. (2007). Handling ignorance in argumentation: Semantics of partial argumentation frameworks. In *Proceedings of the 9th European Conference on Symbolic and Quantitative Approaches to Reasoning and Uncertainty*, pages 259–270. Springer-Verlag *Lecture Notes in Artificial Intelligence #4724*.

Cayrol, C. and Lagasquie-Schiex, M. (2005). Graduality in argumentation. *Journal of Artificial Intelligence Research*, 23:245–297.

Cerutti, F., Gaggl, S., Thimm, M., and Wallner, J. (2017). Foundations of implementations for formal argumentation. *Journal of Logics and their Applications*, 4(8):2623–2706.

Cerutti, F., Giacomin, M., and Vallati, M. (2014). ArgSemSAT: Solving argumentation problems using SAT. *Proceedings of the 5th International Conference on Computational Models of Argument*, 266:455–456.

Cerutti, F., Giacomin, M., and Vallati, M. (2016). Where are we now? State of the art and future trends of solvers for hard argumentation problems. In *Proceedings of the 6th International Conference on Computational Models of Argument*, volume 287 of *Frontiers in Artificial Intelligence and Applications*, pages 207–218.

Charwat, G., Dvořák, W., Gaggl, S., Wallner, J., and Woltran, S. (2015). Methods for solving reasoning problems in abstract argumentation – a survey. *Artificial Intelligence*, 220:28–63.

Chesñevar, C., MacGinnis, J., Modgil, S., Rahwan, I., Reed, C., Simari, G., South, M., Vreeswijk, G., and Willmott, S. (2006). Towards an argument interchange format. *The Knowledge Engineering Review*, 21(4):293–316.

Chesñevar, C., Maguitman, A., and Loui, R. (2000). Logical models of argument. *ACM Computing Surveys*, 32(4):337–383.

Cook, S. (1971). The complexity of theorem-proving procedures. In *Proceedings of the third annual ACM symposium on Theory of computing*, pages 151–158. ACM.

Coste-Marquis, S., Devred, C., Konieczny, S., Lagasquie-Schiex, M., and Marquis, P. (2007). On the merging of Dung's argumentation systems. *Artificial Intelligence*, 171(10):730–753.

Coste-Marquis, S., Konieczny, S., Mailly, J., and Marquis, P. (2015). Extension enforcement in abstract argumentation as an optimization problem. In *Proceedings of the 24th International Joint Conference on Artificial Intelligence*, pages 2876–2882. AAAI Press/IJCAI.

Diller, M., Wallner, J., and Woltran, S. (2015). Reasoning in abstract dialectical frameworks using quantified boolean formulas. *Argument & Computation*, 6(2):149–177.

Diller, M., Wyner, A., and Straß, H. (2017). Defeasible AceRules: A prototype. In *12th International Conference on Computational Semantics.*

Dimopoulos, Y. and Torres, A. (1996). Graph theoretical structures in logic programs and default theories. *Theoretical Computer Science*, 170(1):209–244.

Doutre, S. and Mailly, J. (2018). Constraints and changes: A survey of abstract argumentation dynamics. *Argument & Computation*, 9(3):223–248.

Dung, P. (1995). On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games. *Artificial Intelligence*, 77(2):321–357.

Dung, P., Mancarella, P., and Toni, F. (2006). A dialectic procedure for sceptical, assumption-based argumentation. In *Proceedings of the 1st International Conference on Computational Models of Argument*, volume 144 of *Frontiers in Artificial Intelligence and Applications*, pages 145–156. IOS Press.

Dung, P. and Thang, P. (2010). Towards (probabilistic) argumentation for jury-based dispute resolution. In *Proceedings of the 3rd International Conference on Computational Models of Argument*, volume 216 of *Frontiers in Artificial Intelligence and Applications*, pages 171–182. IOS Press.

Dunne, P. and Bench-Capon, T. (2002). Coherence in finite argument systems. *Artificial Intelligence*, 141(1):187–203.

Dunne, P. and Wooldridge, M. (2009). Complexity of abstract argumentation. In Rahwan, I. and Simari, G., editors, *Argumentation in Artificial Intelligence*, chapter 5, pages 85–104. Springer.

Dvořák, W., Järvisalo, M., Wallner, J., and Woltran, S. (2012). CEGARTIX: A SAT-based argumentation system. In *Proceedings of the Pragmatics of SAT Workshop*.

Egly, U., Gaggl, S., and Woltran, S. (2010). Answer-set programming encodings for argumentation frameworks. *Argument & Computation*, 1(2):147–177.

Ellmauthaler, S. and Straß, H. (2014). The DIAMOND system for computing with abstract dialectical frameworks. In *Proceedings of the 5th International Conference on Computational Models of Argument*, volume 266 of *Frontiers in Artificial Intelligence and Applications*, pages 233–240. IOS Press.

Fuchs, N., Kaljurand, K., and Kuhn, T. (2008). Attempto controlled english for knowledge representation. In *Reasoning Web*, pages 104–124. Springer *Lecture Notes in Computer Science #5224*.

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

Garey, M. and Johnson, D. (1979). *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman and Company.

Gordon, T., Prakken, H., and Walton, D. (2007). The Carneades model of argument and burden of proof. *Artificial Intelligence*, 171(10):875–896.

Heras, S., Atkinson, K., Botti, V., Grasso, F., Julián, V., and McBurney, P. (2010). How argumentation can enhance dialogues in social networks. In *Proceedings of the 3rd International Conference on Computational Models of Argument*, volume 216 of *Frontiers in Artificial Intelligence and Applications*, pages 267–274. IOS Press.

Hunter, A. (2014). Probabilistic qualification of attack in abstract argumentation. *International Journal of Approximate Reasoning*, 55(2):607–638.

Janssen, J., Cock, M., and Vermeir, D. (2008). Fuzzy argumentation frameworks. In *Proceedings of the 12th International Conference on Information Processing and Management of Uncertainty in Knowledge-based Systems*, pages 513–520.

Johnson, D. (1974). Approximation algorithms for combinatorial problems. *Journal of Computer and System Sciences*, 9(3):256–278.

Konczak, K. and Lang, J. (2005). Voting procedures with incomplete preferences. In *Proceedings of the Multidisciplinary IJCAI-05 Workshop on Advances in Preference Handling*, pages 124–129.

Krauthoff, T., Baurmann, M., Betz, G., and Mauve, M. (2016). Dialog-based online argumentation. In *Proceedings of the 6th International Conference on Computational Models of Argument*, volume 287 of *Frontiers in Artificial Intelligence and Applications*, pages 33–40. IOS Press.

Krauthoff, T., Meter, C., Betz, G., Baurmann, M., and Mauve, M. (2018). D-BAS – a dialog-based online argumentation system. In *Proceedings of the 7th International Conference on Computational Models of Argument*, volume 305 of *Frontiers in Artificial Intelligence and Applications*, pages 325–336.

Lang, J., Rey, A., Rothe, J., Schadrack, H., and Schend, L. (2015). Representing and solving hedonic games with ordinal preferences and thresholds. In *Proceedings of the 14th International Conference on Autonomous Agents and Multiagent Systems*, pages 1229–1237. IFAAMAS.

Lawrence, J., Bex, F., Reed, C., and Snaith, M. (2012). AIFdb: Infrastructure for the argument web. In *Proceedings of the 4th International Conference on Computational Models of Argument*, pages 515–516.

Levin, L. (1973). Universal sorting problems. *Problemy Peredachi Informatsii*, 9(3):265–266.

Li, H., Oren, N., and Norman, T. (2011). Probabilistic argumentation frameworks. In *Proceedings of the 1st International Workshop on Theory and Applications of Formal Argumentation*, pages 1–16. Springer-Verlag *Lecture Notes in Artificial Intelligence #7132*.

Liao, B., Jin, L., and Koons, R. (2011). Dynamics of argumentation systems: A division-based method. *Artificial Intelligence*, 175(11):1790–1814.

Linsbichler, T., Maratea, M., Niskanen, A., Wallner, J., and Woltran, S. (2018). Novel algorithms for abstract dialectical frameworks based on complexity analysis of subclasses and SAT solving. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence*, pages 1905–1911.

Meyer, A. and Stockmeyer, L. (1972). The equivalence problem for regular expressions with squaring requires exponential space. In *Proceedings of the 13th IEEE Symposium on Switching and Automata Theory*, pages 125–129. IEEE Computer Society Press.

Modgil, S. and Prakken, H. (2011). Revisiting preferences and argumentation. In *Proceedings of the 22nd International Joint Conference on Artificial Intelligence*, pages 1021–1026.

Modgil, S. and Prakken, H. (2013). A general account of argumentation with preferences. *Artificial Intelligence*, 195:361–397.

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

Neugebauer, D. (2017). Generating defeasible knowledge bases from real-world argumentations using D-BAS. In *Proceedings of the 1st Workshop on Advances In Argumentation in Artificial Intelligence*, volume 2012, pages 105–110. CEUR Workshop Proceedings.

Neugebauer, D. (2018). DABASCO: Generating AF, ADF, and ASPIC$^+$ instances from real-world discussions. In *Proceedings of the 7th International Conference on Computational Models of Argument*, volume 305 of *Frontiers in Artificial Intelligence and Applications*, pages 469–470. IOS Press.

Neugebauer, D. (2019). DABASCO: Generating defeasible theories from real-world discussions using D-BAS. *Argument & Computation*. Submitted.

Oomidou, Y., Katsura, Y., Sawamura, H., Riche, J., and Hagiwara, T. (2014). Asynchronous argumentation system PIRIKA for anyone, anytime, anywhere, with the balanced semantics. In *Proceedings of the 5th International Conference on Computational Models of Argument*, volume 266 of *Frontiers in Artificial Intelligence and Applications*, pages 471–472. IOS Press.

Pang, B. and Lee, L. (2008). Opinion mining and sentiment analysis. *Foundations and Trends in Information Retrieval*, 2(1–2):1–135.

Papadimitriou, C. (1995). *Computational Complexity.* Addison-Wesley, 2nd edition. Reprinted with corrections.

Peldszus, A. and Stede, M. (2013). From argument diagrams to argumentation mining in texts: A survey. *International Journal of Cognitive Informatics and Natural Intelligence*, 7(1):1–31.

Pollock, J. (1970). The structure of epistemic justification. *American Philosophical Quarterly (4)*, pages 62–78.

Pollock, J. (1987). Defeasible reasoning. *Cognitive Science*, 11(4):481–518.

Pollock, J. (1994). Justification and defeat. *Artificial Intelligence*, 67(2):377–407.

Prakken, H. (2010). An abstract framework for argumentation with structured arguments. *Argument & Computation*, 1(2):93–124.

Prakken, H. (2017). Historical overview of formal argumentation. *IfColog Journal of Logics and their Applications*, 4(8):2183–2262.

Prakken, H. and Vreeswijk, G. (2001). Logics for defeasible argumentation. In *Handbook of Philosophical Logic*, volume 4, pages 219–318.

Prasad, M., Biere, A., and Gupta, A. (2005). A survey of recent advances in SAT-based formal verification. *International Journal on Software Tools for Technology Transfer*, 7(2):156–173.

Rahwan, I. (2008). Mass argumentation and the semantic web. *Web Semantics: Science, Services and Agents on the World Wide Web*, 6(1):29–37.

Rahwan, I. and Simari, G., editors (2009). *Argumentation in Artificial Intelligence.* Springer.

Reed, C., Janier, M., and Lawrence, J. (2014). OVA+: An argument analysis interface. In *Proceedings of the 5th International Conference on Computational Models of Argument*, volume 266, pages 463–464.

Reed, C. and Rowe, G. (2004). Araucaria: Software for argument analysis, diagramming and representation. *International Journal on Artificial Intelligence Tools*, 13(4):961–979.

Rienstra, T. (2012). Towards a probabilistic Dung-style argumentation system. In *Proceedings of the 1st International Conference on Agreement Technologies*, pages 138–152. CEUR Workshop Proceedings.

Rothe, J. (2005). *Complexity Theory and Cryptology. An Introduction to Cryptocomplexity.* EATCS Texts in Theoretical Computer Science. Springer-Verlag.

Scheuer, O., Loll, F., Pinkwart, N., and McLaren, B. (2010). Computer-supported argumentation: A review of the state of the art. *International Journal of Computer-Supported Collaborative Learning*, 5(1):43–102.

Schneider, J., Groza, T., and Passant, A. (2013). A review of argumentation for the social semantic web. *Semantic Web*, 4(2):159–218.

Shum, S. B. (2008). Cohere: Towards web 2.0 argumentation. In *Proceedings of the 2nd International Conference on Computational Models of Argument*, volume 8 of *Frontiers in Artificial Intelligence and Applications*, pages 97–108. IOS Press.

Snaith, M., Devereux, J., Lawrence, J., and Reed, C. (2010). Pipelining argumentation technologies. In *Proceedings of the 3rd International Conference on Computational Models of Argument*, volume 216 of *Frontiers in Artificial Intelligence and Applications*, pages 447–453.

Snaith, M. and Reed, C. (2012). TOAST: Online ASPIC$^+$ implementation. In *Proceedings of the 4th International Conference on Computational Models of Argument*, volume 245 of *Frontiers in Artificial Intelligence and Applications*, pages 509–510. IOS Press.

Stockmeyer, L. (1976). The polynomial-time hierarchy. *Theoretical Computer Science*, 3(1):1–22.

Straß, H. (2014). Implementing instantiation of knowledge bases in argumentation frameworks. In *Proceedings of the 5th International Conference on Computational Models of Argument*, volume 266 of *Frontiers in Artificial Intelligence and Applications*, pages 475–476. IOS Press.

Straß, H. (2015). Instantiating rule-based defeasible theories in abstract dialectical frameworks and beyond. *Journal of Logic and Computation*, 28(3):605–627.

Toni, F. and Torroni, P. (2011). Bottom-up argumentation. In *Proceedings of the 1st International Workshop on Theory and Applications of Formal Argumentation*, volume 7132, pages 249–262. Springer-Verlag *Lecture Notes in Artificial Intelligence #7132.*

Turing, A. (1937). On computable numbers, with an application to the Entscheidungsproblem. *Proceedings of the London Mathematical Society*, 42(1):230–265.

van Gelder, T. (2007). The rationale for Rationale. *Law, Probability and Risk*, 6(1-4):23–42.

Verheij, B. (1996). Two approaches to dialectical argumentation: Admissible sets and argumentation stages. In *Proceedings of the 8th Dutch Conference on Artificial Intelligence*, pages 357–368.

Wallner, J., Niskanen, A., and Järvisalo, M. (2016). Complexity results and algorithms for extension enforcement in abstract argumentation. In *Proceedings of the 30th AAAI Conference on Artificial Intelligence*, pages 1088–1094. AAAI Press.

Wyner, A., Bench-Capon, T., Dunne, P., and Cerutti, F. (2015). Senses of argument in instantiated argumentation frameworks. *Argument & Computation*, 6(1):50–72.

Wyner, A., van Engers, T., and Hunter, A. (2016). Working on the argument pipeline: Through flow issues between natural language argument, instantiated arguments, and argumentation frameworks. *Argument & Computation*, 7(1):69–89.

Index

ASPIC$^+$, 8
- argument, 9
  - defeasible, 10
  - firm, 10
  - plausible, 10
  - strict, 10
- argumentation system, 9
- argumentation theory, 9
- knowledge base, 9
- rationality postulate, 13

abstract dialectical framework, 18
- interpretation, 18
- model, 18

ADF, *see* abstract dialectical framework
AF, *see* argumentation framework
algorithm, 20
- determinism, 20
- run time, 20
  - worst-case, 20

argumentation framework, 14
- argument defense, 14
- characteristic function, 14
- credulous acceptance, 17
- extension, 15
- semantics, 15
  - admissible, 14
  - complete, 15
  - conflict-free, 14
  - grounded, 15
  - preferred, 15
  - stable, 15
- skeptical acceptance, 17

clause, 6
complexity class, 21
conjunction, 6
conjunctive normal form, 6
consistency of sets of literals, 5

decision problem, 19
- completeness, 24
- hardness, 24
- instance, 19
- oracle, 22

disjunction, 6
disjunctive normal form, 6

graph, 14

inference, 6
inference rule, 6
- active rule, 6
- defeasible rule, 7
- strict rule, 6

knowledge base, 6

literal, 5

polynomial hierarchy, 22
propositional formula, 5
propositional variable, 5

reducibility, 24

theory, 6
truth assignment, 5
- satisfying assignment, 6

Personal Publications

Reviewed Workshop Papers

Baumeister, D., Neugebauer, D., Rothe, J., and Schadrack, H. (2016). Verification in incomplete argumentation frameworks. In *6th International Workshop on Computational Social Choice*.

Neugebauer, D. (2017). Generating defeasible knowledge bases from real-world argumentations using D-BAS. In *Proceedings of the 1st Workshop on Advances In Argumentation in Artificial Intelligence*, volume 2012, pages 105–110. CEUR Workshop Proceedings.

Baumeister, D., Neugebauer, D., Rothe, J., and Schadrack, H. (2018c). Complexity of verification in incomplete argumentation frameworks. In *7th International Workshop on Computational Social Choice*.

Reviewed Papers published in Conference Proceedings

Baumeister, D., Neugebauer, D., and Rothe, J. (2015b). Verification in attack-incomplete argumentation frameworks. In *Proceedings of the 4th International Conference on Algorithmic Decision Theory*, pages 341–358. Springer-Verlag *Lecture Notes in Artificial Intelligence #9346*.

Baumeister, D., Neugebauer, D., Rothe, J., and Schadrack, H. (2018b). Complexity of verification in incomplete argumentation frameworks. In *Proceedings of the 32nd AAAI Conference on Artificial Intelligence*, pages 1753–1760. AAAI Press.

Baumeister, D., Neugebauer, D., and Rothe, J. (2018a). Credulous and skeptical acceptance in incomplete argumentation frameworks. In *Proceedings of the 7th International Conference on Computational Models of Argument*, volume 305 of *Frontiers in Artificial Intelligence and Applications*, pages 181–192. IOS Press.

Neugebauer, D. (2018). DABASCO: Generating AF, ADF, and ASPIC$^+$ instances from real-world discussions. In *Proceedings of the 7th International Conference on Computational Models of Argument*, volume 305 of *Frontiers in Artificial Intelligence and Applications*, pages 469–470. IOS Press.

Reviewed Journal Articles

Baumeister, D., Neugebauer, D., Rothe, J., and Schadrack, H. (2018d). Verification in incomplete argumentation frameworks. *Artificial Intelligence*, 264:1–26.

Neugebauer, D. (2019). DABASCO: Generating defeasible theories from real-world discussions using D-BAS. *Argument & Computation*. Submitted.

Eidesstattliche Erklärung

entsprechend §5 der Promotionsordnung vom 15.06.2018

Ich versichere an Eides Statt, dass die Dissertation von mir selbständig und ohne unzulässige fremde Hilfe unter Beachtung der „Grundsätze zur Sicherung guter wissenschaftlicher Praxis an der Heinrich-Heine-Universität Düsseldorf“ erstellt worden ist.

Desweiteren erkläre ich, dass ich eine Dissertation in der vorliegenden oder in ähnlicher Form noch bei keiner anderen Institution eingereicht habe.

Ort, Datum	Daniel Neugebauer