Relating Concrete Argumentation Formalisms and Abstract Argumentation

Michael J. Maher
School of Engineering and Information Technology
University of New South Wales, Canberra
ACT 2600, Australia
E-mail: michael.maher@unsw.edu.au

submitted 29 April 2015; accepted 5 June 2015

Abstract

There are a wide variety of formalisms for defeasible reasoning that can be seen as implementing concrete argumentation on defeasible rules. However there has been little work on the relationship between such languages and Dung's abstract argumentation. In this paper we identify two small fragments on which many concrete defeasible formalisms agree. The two fragments are closely related, as we show. The fragments arise as ways to express abstract argumentation frameworks in the concrete formalisms. Our results enable us to transfer complexity lower bounds from abstract argumentation to concrete formalisms.

Introduction

Argumentation and defeasible reasoning are essentially different names for the same thing: resolving conflicting chains of reasoning in a principled way. In modern times, argumentation has been structured through Dung's introduction of abstract argumentation (Dung 1995). Nevertheless, there are very many defeasible reasoning systems that developed in parallel but are not structured – at least not in quite the same way.

An advantage of abstract argumentation is that it abstracts away details of an argument's construction and how conflicting arguments are resolved. As a consequence, results proved on abstract argumentation have the potential to be applicable to a number of concrete instances. On the other hand, this advantage is largely moot for concrete defeasible reasoning systems that are not developed within the abstract argumentation discipline. The problem is that the relationship between abstract argumentation and the individual concrete systems is unclear.

Indeed, there is relatively little work relating abstract argumentation to such concrete defeasible reasoning formalisms, despite the fact that there are clear commonalities. (Dimopoulos and Kakas 1995) defined a logic programming-based formalism inspired by argumentation, and showed it was sound wrt their argumentation semantics. (Governatori et al. 2004) formulated a concrete argumentation system that is equivalent to the defeasible logic $DL(\partial)$. On the other hand, some relations between abstract argumentation and logic programming were already clear in (Dung 1995), and further detail has been added by (Caminada et al. 2015).

In this paper we exploit the common logic programming underpinnings of abstract argumentation and a concrete framework for defeasible reasoning to identify a relationship between the two. We extend this relationship to a wide variety of other concrete defeasible reasoning systems. We demonstrate the usefulness of these results by showing how complexity results for abstract argumentation frameworks can be transferred to concrete systems.

The paper is structured as follows. The next section provides brief background on abstract argumentation, formalisms for defeasible reasoning, and semantics of logic programs. The following section introduces a small fragment of defeasible languages that is capable of expressing abstract argumentation frameworks, and investigates its properties. It is shown that the arguments constructed from this fragment in the ASPIC-style are isomorphic to an abstract argumentation framework. It is also shown that the fragment represents a common core of defeasible reasoning in the sense that a wide range of formalisms for defeasible reasoning agree on the conclusions that should be drawn from such fragments.

The next section establishes general relationships between abstract argumentation frameworks under completist semantics[1] and the fragment in logics in the framework $\mathbf{DL}$. It follows a similar pattern to results in (Caminada et al. 2015) relating argumentation semantics and logic programming semantics.

Then a second small fragment is introduced and shown to be equivalent to the first fragment. Nevertheless, the second fragment is able to be expressed in languages that cannot express the first fragment. It is used to show that several concrete formalisms are able to express the stable semantics of abstract argumentation formalisms.

Finally, there is a brief discussion of the use of these results to transfer complexity results about abstract argumentation to the concrete formalisms. The paper concludes with a summary and brief discussion of future work.

Background

This work involves abstract argumentation in the sense of (Dung 1995), which addresses the evaluation of a set of atomic arguments. An *argumentation framework* $\mathcal{A} = (S, \gg)$ consists of a finite set of arguments S and a binary relation $\gg$ over S, called the attack relation. Roughly, if A is attacked by B then accepting B as justified entails rejecting A. The semantics of an argumentation framework is given in terms of *extensions*, which are subsets of S.

Given an argumentation framework, an argument a is said to be *accepted* in an extension E if $a \in E$, and said to be *rejected* in E if some $b \in E$ attacks a. The set of arguments rejected in E will be denoted E^-. An argument that is neither accepted nor rejected in E is said to be *undecided* in E. An extension E is *conflict-free* if the restriction of $\gg$ to E is empty. An argument a is *defended* by E if every argument that attacks a is attacked by some argument in E. An extension E of $\mathcal{A}$ is *complete* if it is conflict free and, $a \in E$ iff a is defended by E. The smallest complete extension exists and is called the *grounded* extension. The maximal (under the set containment ordering) complete extensions are called the

[1] Completist semantics are semantics defined in terms of complete extensions.

preferred extensions. An extension E of $\mathcal{A}$ is *stable* if it is complete and for every argument $a \in S \backslash E$ there is an argument in E that attacks a. An extension E of $\mathcal{A}$ is *semi-stable* if it is a complete extension such that $E \cup E^-$ is maximal wrt set containment (or, equivalently, the set of undecided arguments is minimal).

Each extension can be considered a potential outcome of evaluating an argumentation framework: classifying arguments as accepted, rejected or undecided. Each class of extensions (complete, preferred, ...) expresses a criterion for an extension to be a "reasonable" outcome; thus each class expresses a semantics of an argumentation framework: the set of reasonable extensions. From these extensions, conclusions about individual arguments can be drawn. Each of these semantics consist only of complete extensions; we call such semantics *completist*.

There are several logical systems that provide for the construction of arguments, which can then be evaluated according to one of the semantics of abstract argumentation. They include the structured argumentation systems ASPIC (Amgoud et al. 2006), ASPIC+ (Prakken 2010) and ASPICLITE (Wu and Podlaszewski 2015), where arguments are essentially proof trees constructed from rules, and assumption-based argumentation (ABA) (Bondarenko et al. 1997), where arguments are sets of assumptions.

However, there are numerous systems for defeasible reasoning that provide concrete mechanisms for drawing conclusions from defeasible rules, without formulating the problem as the construction and then evaluation of arguments. Among such systems are: various systems for non-monotonic inheritance (Touretzky et al. 1987), the defeasible logics NDL and ADL (Maier and Nute 2010), the defeasible logics in the framework of (Antoniou et al. 2000), including the DL and $WFDL$ (Billington et al. 2010; Maher 2013) frameworks, the extended defeasible logics of Billington (for example (Billington 2011)), courteous logic programs (Grosof 1999) and its more recent incarnations LPDA, ASPDA and Rulelog (Wan et al. 2009; Wan et al. 2015; Grosof and Kifer 2013), DEFLOG (Verheij 2003), FDL (Maher 2010), Ordered Logic (Laenens and Vermeir 1990), logic programming without negation as failure (LPwNF) (Dimopoulos and Kakas 1995), and Defeasible Logic Programming (DeLP) (García and Simari 2004).

Common to most of these systems is the expression of defeasible rules and priorities among such rules. Syntax varies in these systems, but we will use $\Rightarrow$ uniformly to express defeasible rules. In general these systems support a variety of additional features such as strict rules, defeaters, conflict sets, mutex, ... Furthermore, many systems admit several variants that treat the interaction of conflicting rules and priorities differently.

Some of these systems draw only positive conclusions, but the defeasible logics draw both positive and negative conclusions. For example, in the defeasible logic $DL(\partial^*)$ we may derive $+\partial^* q$, meaning that q can be proved, or derive $-\partial^* q$, meaning that it can be established within the proof system that q cannot be proved. Of course, it is also possible that neither conclusion can be drawn.

The framework of (Antoniou et al. 2000), which we call $\mathbf{DL}$, will play a central role in this paper. The logics in this framework are determined by two parameters in the framework: a conflict resolution mechanism that specifies the way conflicting rules and priorities interact, and a semantics of logic programming. We will refer to individual logics in this framework as $\mathbf{DL}(t, S)$, where t refers to method of conflict resolution and S refers to a semantics, and we will use $*$ as a wildcard to express classes of logics in $\mathbf{DL}$. Conflict resolution is determined by two key properties: whether ambiguity is propagated or blocked; and whether a single rule must have a higher priority than all conflicting rules in order for it to be applied, or rules can form "teams" that, together, may overrule conflicting rules and allow the rules to be applied. Such issues are important in the application of defeasible rules, but are details that are abstracted away in argumentation. For more on these variants, see (Billington et al. 2010).

Many of the systems identified above can be seen as based on a semantics for negation in logic programs, even though their original formulation was not in those terms. In particular, logics in the DL framework were explicitly shown to employ Kunen's semantics (Kunen 1987) (which, for this paper, is equivalent to Fitting's semantics (Fitting 1985) since we only consider propositional languages and finite theories) while ADL, NDL, courteous logic programs, LPDA, Rulelog, and the logics in $WFDL$ employ the well-founded semantics (Maier and Nute 2010; Wan et al. 2009; Grosof and Kifer 2013; Maher and Governatori 1999; Maher 2014a). Others, such as DEFLOG and ASPDA, reflect the stable model semantics.

The logic programming semantics we will focus on can be seen to be derived from the 3-valued stable models (Przymusinski 1990) (also known as *partial stable models*). In addition to the semantics based on all partial stable models, there is the well-founded model (Gelder et al. 1991), which is the least partial stable model; the (2-valued) stable models (Gelfond and Lifschitz 1988); the regular models (You and Yuan 1994), which are the maximal partial stable models under set inclusion on the positive literals; and the L-stable models (Eiter et al. 1997), which are the maximal partial stable models under set inclusion on positive and negative literals or, equivalently, the minimal partial stable models under set inclusion on the undefined literals. The argumentation semantics defined above are the counterparts of these logic programming semantics (Caminada et al. 2015).

A Small Fragment

We begin by addressing the representation of abstract argumentation frameworks in concrete defeasible reasoning systems. Such systems may have many features, with complex interactions, but only a small fragment of these formalisms is needed to mimic the behaviour of an abstract argumentation framework.

Definition 1
For any abstract argument framework $\mathcal{A}$, the corresponding set of *canonical defeasible rules* $CDR(\mathcal{A})$ is defined as follows:

For each argument A, there is a proposition p_A, and a defeasible rule r'_A:

$$\Rightarrow \neg p_A$$

For each argument A, attacked by arguments $B_1, \ldots, B_n$, there is the corresponding defeasible rule r_A:

$$\neg p_{B_1}, \ldots, \neg p_{B_n} \quad \Rightarrow \quad p_A$$

Finally, we must express that each rule for $\neg p_A$ is overruled by (or has lower priority than) the rule for p_A whenever the latter is applicable. Different concrete systems may express this requirement in different ways, but most commonly it is expressed directly as a relation on rules.

A set of defeasible rules that has the form $CDR(\mathcal{A})$, for some $\mathcal{A}$, is canonical. Canonical defeasible rules are a defeasible rule counterpart of the logic programs defined by (Caminada et al. 2015) to represent argumentation frameworks. Intuitively, they express that A is not accepted unless all its attackers are rejected.

The class of canonical defeasible rules is very simple, involving only defeasible rules and a priority relation on these rules. More complex features that concrete systems might support, such as strict rules, defeaters, conflict sets, mutex, etc. are not present. Hence, a quite wide range of formalisms are able to express canonical defeasible rules.

We first show that a concrete argumentation system applied to arguments constructed from $CDR(\mathcal{A})$ in the ASPIC style comes to the same conclusions as the abstract system.

Define the arguments in the concrete argumentation system to be proof trees constructed from the rules for the propositions p_A and $\neg p_A$, for all arguments A. Hence, the proof trees for p_A are trees with root labelled p_A and n children labelled $\neg p_{B_1}, \ldots, \neg p_{B_n}$ respectively, and proof trees for $\neg p_A$ consist of a single node labelled by $\neg p_A$. Thus arguments have height of at most 1. The attack relation is defined as follows: the argument for p_A is attacked by the argument for p_B iff the argument for $\neg p_B$ is a subargument of the argument for p_A. Such an attack relation arises in any concrete argumentation system that employs undercutting (sometimes called undermining) attacks and can express that each argument for p_A has priority over the argument for $\neg p_A$. If we ignore the arguments for $\neg p_A$, which in any case do not attack any other argument, the concrete argument system is isomorphic to the abstract argumentation framework from which it was derived.

Proposition 2
Let $\mathcal{A}$ be an abstract argumentation framework and consider $CDR(\mathcal{A})$. The concrete argument system derived from $CDR(\mathcal{A})$, restricted to arguments for propositions p_A, is isomorphic to $\mathcal{A}$.

As a result, for every common argumentation semantics, $\mathcal{A}$ and the argumentation framework derived from $CDR(\mathcal{A})$ derive the same conclusions. This result assures us that the canonical defeasible rules accurately represent the argumentation framework.

Among the formalisms that can represent canonical defeasible rules are: the defeasible logics NDL and ADL, the defeasible logics in the **DL** framework, the extended defeasible logics of Billington, courteous logic programs and its more recent incarnations LDPA, ASPDA and Rulelog, Ordered Logic and LPwNF. Similarly, structured argumentation systems in the ASPIC style and assumption-based argumentation can express canonical defeasible rules.

Furthermore, many of these concrete systems infer the same consequences from canonical defeasible rules. Thus canonical defeasible rules form a core on which these defeasible reasoning systems agree.

Theorem 3
The positive conclusions drawn from a set of canonical defeasible rules are the same, whether the rules are interpreted in any of the following formalisms: the defeasible logics

NDL and ADL, the logics in the frameworks DL and $WFDL$, Billington's defeasible logics, and the formalisms Ordered Logic, LPwNF, courteous logic programs and LPDA[2].

Furthermore, for those formalisms that support negative conclusions, the negative conclusions drawn from a set of canonical defeasible rules are also the same.

This result is a consequence of the particularly simple form of canonical defeasible rules: there is one rule for $\neg p_A$ and at most one rule for p_A, which has the higher priority. Consequently, the many variations in how conflicting rules and priorities interact converge on the same behaviour.

There are some defeasible reasoning systems that seem unable to represent canonical defeasible rules. Traditional non-monotonic inheritance systems are unable to express rules with multiple body atoms and rules with negative literals in the body. Logics where priorities cannot be given independently have an obvious problem. Such logics include a defeasible logic in (Nute 1994) where priorities are determined by a specificity relation, and FDL, where priorities are related to length of defeasible derivations. When the language is restricted to defeasible rules (that is, without the ability to express priorities), ambiguity propagating defeasible logics seem unable to represent priorities (Lam and Governatori 2011), but ambiguity blocking defeasible logics can represent them with auxiliary defeasible rules (Antoniou et al. 2001). Finally, DEFLOG is unable to directly express canonical defeasible rules because the dialectical negation in that language overrules a corresponding un-negated proposition; however, if we encode p_A as $\times p_A$ (the dialectical negation of p_A), and encode $\neg p_A$ as p_A then DEFLOG can express these rules.

Relationships

In (Antoniou et al. 2000), logic programs are used as meta-programs to define the way conflicting defeasible rules and priorities on rules interact. A particular logic is characterized by a meta-program and a semantics for logic programs. The semantics is applied to the logic program consisting of the meta-program and facts describing the rules and priorities. The meta-program for logics like $DL(\partial^*)$, pared down to address only defeasible rules and priorities, consists of the following two rules.

```
c1   defeasibly(X):-
         rule(R, X, [Y_1, ..., Y_n]),
         defeasibly(Y_1),...,defeasibly(Y_n),
         not overruled(R, X).

c2   overruled(R, X):-
         rule(S, ~X, [U_1, ..., U_n]),
         defeasibly(U_1),...,defeasibly(U_n),
         not sup(R, S).
```

Here sup(R, S) expresses that the rule R is superior to (i.e. has priority over) the rule S, rule$(R, X, [Y_1, \dots, Y_n])$ represents a defeasible rule called R with head X and body $Y_1, \dots, Y_n$, and $\sim$ expresses negation in the object language. defeasibly(X) expresses

[2] We assume that the LPDA theory has the overriding property (Wan et al. 2009).

that the literal X can be concluded defeasibly, that is, $+\partial^* X$ is a consequence of the object defeasible theory. Similarly, $\neg\texttt{defeasibly}(X)$ expresses $-\partial^* X$. If D denotes a set of rules and priorities, we use $M(D)$ to denote the combination of the meta-program with the representation of the rules and priorities of D.

Using this meta-program, we can establish the relationship between an abstract argumentation framework and ambiguity blocking logics in the $\mathbf{DL}$ framework.

Theorem 4
Let M be the meta-program for $\mathbf{DL}(\partial^*, *)$ or $\mathbf{DL}(\partial, *)$. Let $\mathcal{A}$ be an abstract argumentation framework. Then there is an isomorphism between

- complete extensions of $\mathcal{A}$ and partial stable models of $M(CDR(\mathcal{A}))$
- grounded extension of $\mathcal{A}$ and well-founded model of $M(CDR(\mathcal{A}))$
- stable extensions of $\mathcal{A}$ and stable models of $M(CDR(\mathcal{A}))$
- preferred extensions of $\mathcal{A}$ and regular models of $M(CDR(\mathcal{A}))$
- semi-stable extensions of $\mathcal{A}$ and L-stable models of $M(CDR(\mathcal{A}))$

where we restrict the models to `defeasibly` atoms.

In particular, the conclusions derivable from $\mathcal{A}$ under a semantics S are the same as those derived in the logic $\mathbf{DL}(\partial^*, S')$ from $CDR(\mathcal{A})$, where S' is the semantics in the above theorem corresponding to S.

This result is the counterpart, for defeasible rules in the $\mathbf{DL}$ framework, of Table 5 of (Caminada et al. 2015) relating abstract argumentation frameworks and logic programs. The result does not extend to ambiguity propagating logics in the $\mathbf{DL}$ framework; this fact is discussed in more detail at the end of the following section.

Using the above theorem and Proposition 2, we can extend Theorem 3 to argumentation-based formalisms under the grounded semantics.

Corollary 5
Let $\mathcal{A}$ be an abstract argumentation framework. Then an argument A is accepted in the grounded extension of $\mathcal{A}$ iff the argument for p_A is accepted under the grounded semantics of $CDR(\mathcal{A})$ by ABA or any of the ASPIC variants iff p_A is a positive conclusion from $CDR(\mathcal{A})$ in any of the formalisms mentioned in Theorem 3.

Similarly, A is rejected in the grounded extension of $\mathcal{A}$ iff the argument for p_A is rejected under the grounded semantics of $CDR(\mathcal{A})$ by ABA or any of the ASPIC variants iff p_A is a negative conclusion from $CDR(\mathcal{A})$ in any of the formalisms mentioned in Theorem 3 that draw negative conclusions.

Thus we see that not only can the grounded semantics can be represented by defeasible formalisms based around the well-founded semantics, it can also be represented by those based on the Fitting/Kunen semantics.

An Alternate Fragment

Although the canonical defeasible rules fragment provides a common core for many defeasible languages, there are some languages that cannot express it, while others require significant distortions to represent it (for example, DefLog). This motivates the investigation of a different fragment that can represent abstract argumentation frameworks.

An alternative representation of an abstract argumentation framework is as follows.

Definition 6

For any argument framework $\mathcal{A}$, the corresponding set of *alternative canonical defeasible rules* $ACDR(\mathcal{A})$ is defined as follows:
For each argument A, there is a proposition p_A and a rule r_A:

$$\Rightarrow p_A$$

For each attack by argument B on argument A, there is the corresponding defeasible rule r_{BA}:

$$p_B \quad \Rightarrow \neg p_A$$

Finally, we must express that each rule for $\neg p_A$ overrules (or has higher priority than) the rule for p_A whenever the former is applicable. Different concrete systems may express this requirement in different ways.

This definition is similar to the mapping of argumentation frameworks to logic programs by (Dung 1995). Intuitively, it expresses that A is accepted unless there is an attacker that is accepted. It can be directly expressed in DEFLOG, using the dialectical negation $\times$ in place of the usual negation. The nature of $\times$ ensures that rules for $\times p_A$ overrule the rule for p_A. Indeed, (Verheij 2003) uses this formulation to model argumentation frameworks, and states that dialectical interpretations of such rules correspond to stable extensions. Non-monotonic inheritance networks can also express rules of this form, since the body of rules is a single positive literal, but it is not clear whether the priority relation can be expressed. That would depend on the individual semantics of non-monotonic inheritance.

Again we can construct ASPIC-style arguments from these rules: an argument for p_A consists of the rule r_A, while there may be several arguments for $\neg p_A$, each consisting of r_B and r_{BA}. Every argument for $\neg p_A$ attacks the argument for p_A.

However, the constructed arguments are not isomorphic to $\mathcal{A}$.

Example 7

Consider an abstract argumentation framework $\mathcal{A}$ consisting of arguments A, B, and C, where A attacks both B and C. The arguments constructed from $ACDR(\mathcal{A})$ are arguments for each of the literals p_A, p_B, p_C, $\neg p_B$ and $\neg p_C$. Then the argument for $\neg p_B$ attacks the argument for p_B, and similarly for C, but the argument for p_A is not involved in an attack. Furthermore, there is no single argument that attacks both p_B and p_C, the way A attacks both B and C in $\mathcal{A}$. Hence the constructed arguments are not isomorphic to $\mathcal{A}$.

Nevertheless, the alternative canonical defeasible rules do characterize the conclusions of an argumentation framework under the semantics we consider, as we now establish. First, we need two lemmas.

Lemma 8

Consider the transformation of logic programs which replaces a rule

$$A \quad \text{:-} \quad not\ B_1, \ldots, notB_n.$$

by the rules

$$\begin{array}{lll} A & \text{:-} & not\, C. \\ C & \text{:-} & B_1. \\ & \ldots & \\ C & \text{:-} & B_n. \end{array}$$

where C is a new symbol. Let P be the original program and P' be the transformed program. Then P and P' have the same partial stable models, ignoring C.

Applying the above transformation and other transformations to $M(CDR(\mathcal{A}))$, we can obtain $M(ACDR(\mathcal{A}))$ (up to renaming of introduced symbols), where M is a meta-program for an ambiguity blocking logic in $\mathbf{DL}$. Thus

Lemma 9
Let M be the meta-program for an ambiguity blocking logic in the $\mathbf{DL}$ framework. $M(CDR(\mathcal{A}))$ and $M(ACDR(\mathcal{A}))$ have the same partial stable models, when restricted to literals involving `defeasibly`.

Consequently, Theorem 4 also applies to the alternate canonical defeasible rules.

Theorem 10
Let M be a meta-program for an ambiguity blocking logic in the $\mathbf{DL}$ framework. Let $\mathcal{A}$ be an abstract argumentation framework. Then there is an isomorphism between

- complete extensions of $\mathcal{A}$ and partial stable models of $M(ACDR(\mathcal{A}))$
- grounded extension of $\mathcal{A}$ and well-founded model of $M(ACDR(\mathcal{A}))$
- stable extensions of $\mathcal{A}$ and stable models of $M(ACDR(\mathcal{A}))$
- preferred extensions of $\mathcal{A}$ and regular models of $M(ACDR(\mathcal{A}))$
- semi-stable extensions of $\mathcal{A}$ and L-stable models of $M(ACDR(\mathcal{A}))$

where we restrict the models to `defeasibly` atoms.

Similarly, we can obtain a counterpart of Theorem 3 for $ACDR$.

Furthermore, we can establish a similar result for the stable semantics. Relatively few concrete defeasible languages support the stable semantics, although the $\mathbf{DL}$ framework supported this semantics and recently further proposals have been made (Maier 2013; Wan et al. 2015). ASPDA (Wan et al. 2015), like LPDA, allows the interaction between conflicting rules to be defined in the theory. We restrict attention to theories satisfying the overriding property (Wan et al. 2009) to avoid some nonsensical theories. (Maier 2013) extends ADL and NDL in the style of the stable model semantics to give α-stable sets (extending ADL) and β-stable sets (extending NDL).

Theorem 11
Let M be a meta-program for an ambiguity blocking logic in the $\mathbf{DL}$ framework. Let $\mathcal{A}$ be an abstract argumentation framework. Then there are isomorphisms between

- stable extensions of $\mathcal{A}$
- dialectical interpretations of $ACDR(\mathcal{A})$ in DEFLOG
- stable models of $M(ACDR(\mathcal{A}))$ restricted to `defeasibly` atoms
- stable models of $M(CDR(\mathcal{A}))$ restricted to `defeasibly` atoms

- stable models of $CDR(\mathcal{A})$ in ASPDA
- stable models of $ACDR(\mathcal{A})$ in ASPDA
- β-stable sets of $CDR(\mathcal{A})$
- β-stable sets of $ACDR(\mathcal{A})$

where we assume that the ASPDA theory has the overriding property.

The proof uses results of (Verheij 2003; Dung 1995; Wan et al. 2015) and Theorems 4 and 10.

Theorem 4 and the results in this section apply only to the ambiguity blocking logics in $\mathbf{DL}$; they do not extend to the ambiguity propagating logics. The following example demonstrates the situation.

Example 12
Consider an argumentation framework $\mathcal{A}$ with two arguments, A and B, where A attacks B and B attacks A. Under the stable semantics there are two stable extensions: one in which A is accepted and B is rejected, and one in which B is accepted and A is rejected.

The canonical defeasible rules corresponding to $\mathcal{A}$ are

$$\begin{array}{rcr} & \Rightarrow & \neg p_A \\ \neg p_B & \Rightarrow & p_A \\ & \Rightarrow & \neg p_B \\ \neg p_A & \Rightarrow & p_B \end{array}$$

After substantial simplification, the application of the meta-program for $\mathbf{DL}(\delta^*, *)$ – an ambiguity propagating logic – results in the following rules, among others:

$$\begin{array}{rcl} \texttt{defeasibly}(p_A) & :- & \textit{not}\ \texttt{support}(p_B) \\ \texttt{defeasibly}(p_B) & :- & \textit{not}\ \texttt{support}(p_A) \\ \texttt{support}(p_A) & :- & \textit{not}\ \texttt{defeasibly}(p_B) \\ \texttt{support}(p_B) & :- & \textit{not}\ \texttt{defeasibly}(p_A) \end{array}$$

These rules admit a stable model containing $\{\texttt{defeasibly}(p_A), \texttt{defeasibly}(p_B)\}$. Consequently, the stable models of $M(CDR(\mathcal{A}))$ are not isomorphic to the stable extensions of $\mathcal{A}$.

For comparison, the corresponding logic program for $\mathbf{DL}(\partial^*, *)$ contains

$$\begin{array}{rcl} \texttt{defeasibly}(p_A) & :- & \textit{not}\ \texttt{defeasibly}(p_B) \\ \texttt{defeasibly}(p_B) & :- & \textit{not}\ \texttt{defeasibly}(p_A) \end{array}$$

which has the two stable models corresponding the stable extensions of $\mathcal{A}$.

The isomorphism fails because, in the meta-program for ambiguity propagating logics in $\mathbf{DL}$, two mutually recursive predicates – `defeasibly` and `support` – are used, rather than a single predicate. As a result, there is no dependency relation between $\texttt{defeasibly}(p_A)$ and $\texttt{defeasibly}(p_B)$ corresponding to the attack relation between A and B.

Complexity

The results in this paper allow us to transfer complexity lower bounds for problems on abstract argumentation frameworks to the corresponding concrete formalisms.

For example, consider the problem of adding additional arguments to an argumentation framework (called expansion by (Baumann and Brewka 2010)) so that a specified argument is accepted under the grounded semantics of the revised argumentation framework. This is a form of abduction (Booth et al. 2014; Maher 2014b), and it arises in strategic argumentation (Governatori et al. 2014). This problem can be shown to be NP-hard for abstract argumentation frameworks by reduction of 3SAT. It then follows that the corresponding abduction problem is also NP-hard in all the formalisms mentioned in Theorem 3 and Corollary 5. This shortcuts proofs of results in (Governatori et al. 2014; Maher 2014b; Governatori et al. 2014), and establishes several new results.

In general, concrete systems have extra features – beyond defeasible rules and priorities – that can add extra complexity to inference in those systems. Consequently, tight complexity lower bounds at the abstract level do not necessarily imply tight bounds for a concrete system.

Conclusions and Future Work

We have identified two complementary fragments of defeasible rule systems that are each capable of representing abstract argumentation frameworks. We showed that canonical defeasible rules represent a core of defeasible reasoning in that a wide variety of concrete formalisms agree on the meaning of this fragment. In doing so, we demonstrated that the majority of concrete formalisms for defeasible reasoning reflect the grounded semantics of argumentation.

We showed that several completist semantics for abstract argumentation are represented in the $\mathbf{DL}$ framework under various corresponding logic programming semantics. Several concrete formalisms, including $\mathbf{DL}(\partial^*)$ under the stable model semantics, were shown to reflect the stable semantics of argumentation.

Some formalisms, notably non-monotonic inheritance, have no obvious way to represent abstract argumentation frameworks. Despite the introduction of the second fragment, which is more amenable to the syntactic restrictions of these formalisms, there is no direct way to represent priorities. Some semantics of inheritance might permit the encoding of priorities so it would be interesting, for completeness, to understand the status of the different semantics of non-monotonic inheritance.

Theorem 3 and Corollary 5 apply to both ambiguity blocking and ambiguity propagating formalisms. However, Theorems 4 and 10 apply only to the ambiguity blocking logics in the $\mathbf{DL}$ framework. These results do not extend to the ambiguity propagating logics in the framework, but this appears to be a consequence of how they are represented in the $\mathbf{DL}$ framework, rather than a reflection of ambiguity propagation. Similarly, the formalisms in Theorem 11 are ambiguity blocking. It will be interesting to see whether the α-stable sets (the extension of ADL, an ambiguity propagating logic) (Maier 2013) participate in the isomorphisms of Theorem 11.

It seems likely that semantics of logic programs can be designed to correspond to other completist argumentation semantics, such as the ideal (Dung et al. 2007) and eager (Caminada 2007) semantics. In that case, Theorems 4 and 10 will extend to such semantics.

References

AMGOUD, L., BODENSTAFF, L., CAMINADA, M., MCBURNEY, P., PARSONS, S., PRAKKEN, H., VAN VEENEN, J., AND VREESWIJK, G. 2006. Final review and report on formal argumentation system. Tech. rep.

ANTONIOU, G., BILLINGTON, D., GOVERNATORI, G., AND MAHER, M. J. 2000. A flexible framework for defeasible logics. In *AAAI/IAAI*. AAAI Press / The MIT Press, 405–410.

ANTONIOU, G., BILLINGTON, D., GOVERNATORI, G., AND MAHER, M. J. 2001. Representation results for defeasible logic. *ACM Trans. Comput. Log. 2,* 2, 255–287.

BAUMANN, R. AND BREWKA, G. 2010. Expanding argumentation frameworks: Enforcing and monotonicity results. In *COMMA*. 75–86.

BILLINGTON, D. 2011. A defeasible logic for clauses. In *AI 2011: Advances in Artificial Intelligence*. Lecture Notes in Computer Science, vol. 7106. Springer, 472–480.

BILLINGTON, D., ANTONIOU, G., GOVERNATORI, G., AND MAHER, M. J. 2010. An inclusion theorem for defeasible logics. *ACM Trans. Comput. Log. 12,* 1, 6.

BONDARENKO, A., DUNG, P. M., KOWALSKI, R. A., AND TONI, F. 1997. An abstract, argumentation-theoretic approach to default reasoning. *Artif. Intell. 93*, 63–101.

BOOTH, R., GABBAY, D. M., KACI, S., RIENSTRA, T., AND VAN DER TORRE, L. W. N. 2014. Abduction and dialogical proof in argumentation and logic programming. In *ECAI 2014 - 21st European Conference on Artificial Intelligence*. 117–122.

CAMINADA, M. 2007. Comparing two unique extension semantics for formal argumentation: Ideal and eager. In *Proc. of the 2007 Benelux Conf. on Artificial Intelligence*. 81–87.

CAMINADA, M., SÁ, S., ALCÂNTARA, J., AND DVORÁK, W. 2015. On the equivalence between logic programming semantics and argumentation semantics. *Int. J. Approx. Reasoning 58*, 87–111.

DIMOPOULOS, Y. AND KAKAS, A. C. 1995. Logic programming without negation as failure. In *Proceedings of the 1995 International Symposium on Logic Programming*. 369–384.

DUNG, P. M. 1995. On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games. *Artif. Intell. 77,* 2, 321–358.

DUNG, P. M., MANCARELLA, P., AND TONI, F. 2007. Computing ideal sceptical argumentation. *Artif. Intell. 171,* 10-15, 642–674.

EITER, T., LEONE, N., AND SACCÀ, D. 1997. On the partial semantics for disjunctive deductive databases. *Ann. Math. Artif. Intell. 19,* 1-2, 59–96.

FITTING, M. 1985. A Kripke-Kleene semantics for logic programs. *J. Log. Program. 2,* 4, 295–312.

GARCÍA, A. J. AND SIMARI, G. R. 2004. Defeasible logic programming: An argumentative approach. *TPLP 4,* 1-2, 95–138.

GELDER, A. V., ROSS, K. A., AND SCHLIPF, J. S. 1991. The well-founded semantics for general logic programs. *J. ACM 38,* 3, 620–650.

GELFOND, M. AND LIFSCHITZ, V. 1988. The stable model semantics for logic programming. In *Logic Programming, Proceedings of the Fifth International Conference and Symposium, Seattle, Washington, August 15-19, 1988 (2 Volumes)*. 1070–1080.

GOVERNATORI, G., MAHER, M. J., ANTONIOU, G., AND BILLINGTON, D. 2004. Argumentation semantics for defeasible logic. *J. Log. Comput. 14,* 5, 675–702.

GOVERNATORI, G., MAHER, M. J., OLIVIERI, F., SCANNAPIECO, S., AND ROTOLO, A. 2014. The complexity of strategic argumentation under grounded semantics. In *Proc. European Conf. on Multi-Agent Systems*. 379–387.

GOVERNATORI, G., OLIVIERI, F., SCANNAPIECO, S., ROTOLO, A., AND CRISTANI, M. 2014. Strategic argumentation is NP-complete. In *Proc. European Conf. on Artificial Intelligence*. 399–404.

GROSOF, B. AND KIFER, M. 2013. Rulelog: Syntax and semantics. http://ruleml.org/rif/rulelog/spec/Rulelog.html. Accessed: April 2015.

GROSOF, B. N. 1999. Compiling prioritized default rules into ordinary logic programs. Tech. rep., IBM.

KUNEN, K. 1987. Negation in logic programming. *J. Log. Program. 4,* 4, 289–308.

LAENENS, E. AND VERMEIR, D. 1990. A fixpoint semantics for ordered logic. *J. Log. Comput. 1,* 2, 159–185.

LAM, H.-P. AND GOVERNATORI, G. 2011. What are the necessity rules in defeasible reasoning? In *LPNMR*. Lecture Notes in Computer Science, vol. 6645. Springer, 187–192.

MAHER, M. J. 2010. Human and unhuman commonsense reasoning. In *LPAR (Yogyakarta)*. 16–29.

MAHER, M. J. 2013. Relative expressiveness of well-founded defeasible logics. In *Proc. Australasian Joint Conf. on Artificial Intelligence*. 338–349.

MAHER, M. J. 2014a. Comparing defeasible logics. In *Proc. European Conf. on Artificial Intelligence*. 585–590.

MAHER, M. J. 2014b. Complexity of exploiting privacy violations in strategic argumentation. In *Proc. Pacific Rim International Conf. on Artificial Intelligence*. 523–535.

MAHER, M. J. AND GOVERNATORI, G. 1999. A semantic decomposition of defeasible logics. In *AAAI/IAAI*. AAAI Press, 299–305.

MAIER, F. 2013. Interdefinability of defeasible logic and logic programming under the well-founded semantics. *TPLP 13*, 107–142.

MAIER, F. AND NUTE, D. 2010. Well-founded semantics for defeasible logic. *Synthese 176,* 2, 243–274.

NUTE, D. 1994. Defeasible logic. In *Handbook of Logic in Artificial Intelligence and Logic Programming, Vol. III*, D. Gabbay, C. Hogger, and J. Robinson, Eds. Oxford University Press, 353–395.

PRAKKEN, H. 2010. An abstract framework for argumentation with structured arguments. *Argument and Computation 1*, 93–124.

PRZYMUSINSKI, T. C. 1990. The well-founded semantics coincides with the three-valued stable semantics. *Fundam. Inform. 13,* 4, 445–463.

TOURETZKY, D. S., HORTY, J. F., AND THOMASON, R. H. 1987. A clash of intuitions: The current state of nonmonotonic multiple inheritance systems. In *IJCAI*. 476–482.

VERHEIJ, B. 2003. DefLog: on the logical interpretation of prima facie justified assumptions. *J. Log. Comput. 13,* 3, 319–346.

WAN, H., GROSOF, B. N., KIFER, M., FODOR, P., AND LIANG, S. 2009. Logic programming with defaults and argumentation theories. In *ICLP*, P. M. Hill and D. S. Warren, Eds. Lecture Notes in Computer Science, vol. 5649. Springer, 432–448.

WAN, H., KIFER, M., AND GROSOF, B. N. 2015. Defeasibility in answer set programs with defaults and argumentation rules. *Semantic Web 6,* 1, 81–98.

WU, Y. AND PODLASZEWSKI, M. 2015. Implementing crash-resistance and non-interference in logic-based argumentation. *J. Log. Comput. 25,* 2, 303–333.

YOU, J. AND YUAN, L. 1994. A three-valued semantics for deductive databases and logic programs. *J. Comput. Syst. Sci. 49,* 2, 334–361.