Explaining Non-Acceptability in Abstract Argumentation

Zeynep G. Saribatur and **Johannes P. Wallner** and **Stefan Woltran**[1]

Abstract. Argumentation frameworks (AFs) provide a central approach to perform reasoning in many formalisms within argumentation in Artificial Intelligence (AI). Semantics for AFs specify criteria under which sets of arguments can be deemed acceptable, with the notion of admissibility being at the core of main semantics for AFs. A fundamental reasoning task is to find an admissible set containing a queried argument, called credulous acceptance under admissibility. While such a set explains how to argue in favour of a queried argument, finding an explanation in the negative case, i.e., answering why a queried argument is not credulously accepted under admissibility, is less immediate. In this paper, we approach this problem by considering subframeworks of a given AF as witnesses for non-acceptability. Due to the non-monotonicity of semantics for AFs, this requires that every expansion of the witnessing subframework must preserve non-acceptance of the argument—otherwise the subframework would not give sufficient reason for rejection. Among our main contributions (i) we show that this notion of witnessing subframeworks is connected to strong admissibility of AFs, (ii) we investigate the complexity of finding small such subframeworks, and (iii) we extend a recently proposed framework for abstraction in the declarative answer set programming paradigm in order to compute rejecting subframeworks. The resulting system is thus able to deliver explanations also in the case of non-acceptance and we provide a first empirical study that shows the feasibility of our approach.

1 INTRODUCTION

The study of formal argumentation has established itself as a foundational cornerstone of Artificial Intelligence with continuous attention from the research community, and with multiple heterogeneous applications domains, such as legal reasoning, medical sciences, and e-government tools [1]. Central to argumentation within AI are formal approaches to represent and reason on arguments, with argumentation frameworks (AFs) [20] constituting a key approach in this area. AFs provide a clear and crisp reasoning engine, which is at the core of several formalisms in argumentation [3, 8, 32, 7]. Reasoning via AFs is carried out by instantiating arguments and directed conflicts (attacks) between them. Importantly, viewing these arguments as abstract entities and using argumentation semantics, i.e., criteria for finding acceptable sets of arguments, suffices to reason in an argumentative way for a variety of application scenarios.

A crucial ingredient for main argumentation semantics is that of admissible sets of arguments, which represent a conflict-free viewpoint that defends itself against counterarguments. An important reasoning task is to check whether an argument is part of at least one admissible set (called credulous acceptance under admissibility), which indicates one argumentative point of view under which that argument is defendable. If such an admissible set exists, that set provides a direct argumentation witnessing acceptability of the queried argument.

Computationally speaking, while checking existence of an admissible set containing a specified argument is NP-complete in general [22], state-of-the-art systems exist that are capable of finding such admissible sets even when faced with large AFs [16, 40, 27]. However, if there is no admissible set containing a queried argument, i.e., when the argument is not credulously accepted under admissibility, current systems are only able of stating a negative answer without detailed information. That is, current implementation approaches can provide an argumentative witness to credulous acceptability, but no explanations as to why an argument is not credulously accepted.

From the complexity of the underlying problem, it follows that asking whether an argument is not credulously accepted under admissibility is coNP-complete, which implies that witnesses for non-acceptance, under complexity theoretic assumptions, cannot be polynomially bound in size, in general. Nevertheless, in this paper we argue that witnesses can be produced that provide reasonable and potentially small explanations to non-acceptability of arguments.

Example 1. *Consider a simple AF with four arguments, a, b, c, and d, with a attacking b that attacks c, which in turn attacks d (Figure 1). An admissible set requires that no two arguments in that set are conflicting, and that each attack onto the set is defended against, by an attack on the attacker, from inside the set. In this example, the unique subset maximal admissible set is* $\{a, c\}$*, i.e.,* a *is not doubted at all (no incoming attack) and defends* c *against the attack from* b*.*

$$\begin{tikzcd} a \arrow[r] & b \arrow[r] & c \arrow[r] & d \end{tikzcd}$$

Figure 1. Example AF

Say we desire to reason why d in the example is not credulously accepted, i.e., why, for this AF, no admissible set containing d exists. We approach this question by investigating what represents a sufficient reason to explain non-acceptance, given the AF and the criteria for admissibility.

To answer this question from a structural point of view, we have at our disposal a directed graph of arguments (vertices) and attacks (directed edges). A natural way towards what constitutes as a part of the AF that is sufficient for rejection is to consider which subframework (subgraph) can suffice to show rejection of d. Consider subframeworks induced by subsets of the arguments. The subgraph induced by only c and d is not sufficient for non-acceptance of d. This is because one can add argument b, resulting in an induced subgraph of $\{b, c, d\}$, which leads to d being reinstated by b defending d. However, taking both arguments a and c is enough reason to re-

[1] Institute of Logic and Computation, TU Wien, Vienna, Austria, email: zeynep@kr.tuwien.ac.at, {wallner,woltran}@dbai.tuwien.ac.at

ject d: when faced with a and c neither re-adding b, d, nor both b and d give a reason for accepting d. In this way, the subframework with arguments a and c provides a, what we call, *strong* reason for rejection of d, since no induced subframework in between $\{a, c\}$ to $\{a, b, c, d\}$ (the full framework) accepts d credulously under admissibility. Considering all frameworks in between is vitally important, due to non-monotonicity of AFs. Otherwise, non-monotonic effects, such as adding b to the induced subgraph of c and d, may lead to an unjustified credulous acceptance of d.

The approach of considering subframeworks and all frameworks in between up to the original framework is founded in recent research: indeed, previous work by Brewka, Thimm, and Ulbricht [11] investigated *strong inconsistency* in non-monotonic reasoning using similar concepts. While our notion is inspired and uses similar definitions from their work, their results (which mainly concern consistency of formalisms) do not directly yield results for studying credulous non-acceptance for AFs.

In this paper, we investigate the concept of strong reasons for rejection for credulous non-acceptance, focusing on a structural viewpoint, but also argue that the underlying concept is flexible to also cover different ways of explaining non-acceptance. More concretely, our main contributions are summarized as follows.

- We formally define strongly rejecting subframeworks on AFs that are specified as subframeworks s.t. each superframework (including the subframework itself) does not credulously accept a queried argument under a given semantics. In this paper we focus on the central semantic concept of admissibility, but also discuss and give results for stable and grounded semantics, thus covering all main semantics in terms of credulous acceptance.
- In addition to exemplifying several cases for strongly rejecting subframeworks, we show that our notion also connects to strongly admissible sets [4], in case such sets attack the queried argument. Interestingly, strongly admissible sets were shown to provide a particularly strong game-theoretic notion of showing acceptability. In turn, being attacked by a strongly admissible set presents itself naturally to be a strong explanation of non-acceptability.
- We argue that the underlying concept of strongly rejecting subframeworks is not tied to a single structural view. In particular, we exemplify that similar ideas can be applied to variants of subframeworks, e.g., by relaxing attacks, and also to relaxing semantical concepts by weakening defense of admissible sets. Such variants present different notions of explaining non-acceptability.
- We show that the complexity of finding small strongly rejecting subframeworks is in Σ_2^P, and NP and coNP hard.
- In order to compute strong reasons for rejection, we show that answer set programming (ASP) and a recently proposed framework of abstractions in ASP [37, 38] provide a versatile approach able to capture various forms of strong rejections by a notion called blocker sets. We extend the ASP abstraction framework by a novel approach to compute blocker sets of minimum cardinality, which can be applied also to other domains than argumentation.
- We present an empirical evaluation of a prototype implemented in ASP to compute blocker sets of minimum cardinality, based on instances of the recent ICCMA'19 argumentation competition.

2 ARGUMENTATION FRAMEWORKS

We start the formal preliminaries with argumentation frameworks (AFs) [20] and semantics for AFs [2]. An AF consists of a set of abstract arguments and directed attacks between these arguments.

Definition 1. *An* argumentation framework (AF) *is a pair $F = (A, R)$, where A is a finite set of arguments and $R \subseteq A \times A$ is the attack relation. The pair $(a, b) \in R$ means that a attacks b.*

Example 2. *AFs have a natural representation as directed graphs. Consider an AF $F = (\{a, b, c, d\}, R)$ with four arguments and the following attacks: $R = \{(a, b), (b, c), (c, d)\}$ (Figure 1).*

A central notion towards the semantics of AFs is that of defense of arguments.

Definition 2. *Let $F = (A, R)$ be an AF. An argument $a \in A$ is* defended *(in F) by a set $S \subseteq A$ if for each $b \in A$ such that $(b, a) \in R$ there exists a $c \in S$ such that $(c, b) \in R$.*

Semantics for argumentation frameworks are defined through a function σ which assigns to each AF $F = (A, R)$ a set $\sigma(F) \subseteq 2^A$ of extensions. We consider for σ the functions *cf*, *adm*, *com*, *grd*, *stb*, and *prf*, which stand for conflict-free, admissible, complete, grounded, stable, and preferred, respectively. Towards the definition we make use of the characteristic function of AFs, defined for an AF $F = (A, R)$ by $\mathcal{F}_F(S) = \{x \in A \mid x \text{ is defended by } S\}$.

Definition 3. *Let $F = (A, R)$ be an AF. An $S \subseteq A$ is conflict-free (in F) if there are no $a, b \in S$ such that $(a, b) \in R$. We denote the set of conflict-free sets of F by $cf(F)$. For an $S \in cf(F)$ it holds that*

- *$S \in adm(F)$ iff $S \subseteq \mathcal{F}_F(S)$;*
- *$S \in com(F)$ iff $S = \mathcal{F}_F(S)$;*
- *$S \in grd(F)$ iff S is the least fixed-point of $\mathcal{F}_F$;*
- *$S \in stb(F)$ iff S attacks in F each $a \in A \setminus S$; and*
- *$S \in prf(F)$ iff $S \in adm(F)$ and $\nexists T \in adm(F)$ with $S \subset T$.*

An argument a is said to be credulously accepted under semantics σ in an AF F if there is a σ-extension $E \in \sigma(F)$ such that $a \in E$. An argument a is skeptically accepted under semantics σ in an AF F if for all σ-extensions $E \in \sigma(F)$ it holds that $a \in E$. We denote the set of credulously accepted arguments in F under σ as $cred(F, \sigma) = \{a \mid a \text{ credulously accepted under } \sigma \text{ in } F\}$. It holds that $cred(F, adm) = cred(F, com) = cred(F, prf)$.

3 STRONGLY REJECTING FRAMEWORKS

In this section we formally introduce strongly rejecting (sub)frameworks, and furthermore devote this section to exemplification, study of formal properties, and we show relations of such reasons for non-acceptance. Towards the definition, we first recall a standard notion of subframeworks.

Definition 4. *Let $F = (A, R)$ be an AF, and $S \subseteq A$. Define $F|_S = (S, R')$, with $R' = R \cap (S \times S)$, as the subframework of F w.r.t. S.*

The main definition of strongly rejecting frameworks follows next.

Definition 5. *Let $F = (A, R)$ be an AF, $S \subseteq A$, and $a \in A$ an argument. The subframework $F|_S$ strongly rejects a (w.r.t. F), under semantics σ, if for each S', $S \subseteq S' \subseteq A$, we have $a \notin cred_\sigma(F|_{S'})$.*

In words, a subframework (specified via S) strongly rejects an argument a, under a semantics σ, if this argument is not credulously accepted under σ in the subframework induced by S and *each* subframework "in between" S and the full framework A. This means that $F|_S$ rejects a credulously under σ, and no addition of arguments to S (until the original framework F is reached) can lead to credulous acceptance of a. In particular, this definition excludes the possibility

$$\begin{tikzcd}
a \arrow[r] & b \arrow[r] & c \arrow[dr] & & & \\
 & & & g \arrow[r] & h \arrow[r] & i \\
d \arrow[r] & e \arrow[r] & f \arrow[ur] & & &
\end{tikzcd}$$

(a)

$$\begin{tikzcd}
a \arrow[r] \arrow[dd, leftrightarrow] & b \arrow[r] & c \arrow[dr] & \\
 & & & g \\
d \arrow[r] & e \arrow[r] & f \arrow[ur] &
\end{tikzcd}$$

(b)

$$\begin{tikzcd}
a \arrow[dr] \arrow[dd, leftrightarrow] & & & \\
 & c \arrow[r, leftrightarrow] & d \arrow[r] & e \\
b \arrow[ur] & & &
\end{tikzcd}$$

(c)

$$\begin{tikzcd}
c_1 \arrow[dr] \arrow[drr, bend left] & & \\
\vdots & a \arrow[r] & b \\
c_n \arrow[ur] \arrow[urr, bend right] & &
\end{tikzcd}$$

(d)

Figure 2. Under admissibility, (a) $F|_{\{a,c,h\}}$ and $F|_{\{d,f,h\}}$ strongly reject i, (b) $F|_{\{a,c,f\}}$ and $F|_{\{c,d,f\}}$ strongly reject g, (c) ideal set $\{d\}$ does not strongly reject e, and (d) $F|_{\{a\}}$ strongly rejects b.

that argument a is not credulously accepted in $F|_S$, credulously accepted in a framework $F|_{S'}$ with $S \subseteq S' \subseteq A$ and possibly again rejected in F. Thus, we make sure that non-monotonic effects cannot lead to acceptance: any possible addition must reject a, as well.

Example 3. *Consider an AF $F = (A, R)$ with $A = \{a, b, c, d, e, f, g, h, i\}$ and attacks as in Figure 2(a). Let us have a look at strongly rejecting subframeworks for argument i under admissibility. First, i is not credulously accepted under admissibility in F, since the grounded extension is $\{a, c, d, f, h\}$, which attacks i. This means that $F|_A = F$ is strongly rejecting i under admissibility. Further, if $F|_S$ strongly rejects i under admissibility, then $h \in S$. If $h \notin S$, then $F|_{S\cup\{i\}}$ accepts i credulously under adm, since i is unattacked in this subframework. Argument sets S s.t. $F|_S$ strongly rejects i under adm are, e.g., $\{a, c, h\}$ and $\{d, f, h\}$. Let $S = \{a, c, h\}$. To see that $F|_S$ strongly rejects i under adm, consider any S' with $S' \subseteq (A \setminus S)$: addition of S' yielding $F|_{S\cup S'}$ does not lead to credulous acceptance of i. This holds, because any addition by S' is either attacked by S or attacks arguments not influencing acceptance of i.*

The preceding example suggests several properties of strongly rejecting subframeworks, which we formalize next. First, by definition, strongly rejecting subframeworks enjoy monotonicity.

Proposition 1. *Let $F = (A, R)$ be an AF, σ a semantics, $a \in A$, and $S \subseteq S' \subseteq A$. If $F|_S$ strongly rejects a under σ, then $F|_{S'}$ strongly rejects a under σ.*

That is, if F_S strongly rejects a then each superframework (up to the original one) likewise strongly rejects a. Further basic properties exhibited in Example 3 are presented in the following. In particular, if $F|_S$ strongly rejects a under a semantics we consider here, then a need not be part of S, since a is not credulously accepted in a (sub)framework where the argument does not exist. Further, excluding self-attacking arguments $(a, a) \in R$, at least one attacker of a needs to be in S if $F|_S$ is strongly rejecting under admissibility or grounded semantics. However, if a attacks itself, then even $F|_\emptyset$ strongly rejects; simply because adding a induces addition of (a, a) and any conflict-free semantics does not warrant a, in such a case.

Proposition 2. *Let $F = (A, R)$ be an AF, $a \in A$ an argument, and $\sigma \in \{adm, stb, grd\}$.*

- *If $F|_S$ strongly rejects a under σ, then $F|_{S\setminus\{a\}}$ strongly rejects a under σ.*
- *If $F|_S$ strongly rejects a under adm or grd, and $(a, a) \notin R$, then $S \cap \{b \mid (b, a) \in R\} \neq \emptyset$.*
- *It holds that $F|_\emptyset$ strongly rejects a under σ iff $(a, a) \in R$.*

We note that item two in the preceding proposition is not stated for stable semantics. Indeed, under stable semantics an attacker is not necessarily required. Consider two arguments a and b, with b self-attacking. Then a is not credulously accepted (since the self-attacking b prevents existence of stable extensions), and $F|_{\{b\}}$ strongly rejects a under stable semantics.

While Example 3 might suggest a (direct) connection of admissible sets and (subset minimal) strongly rejecting subframeworks under admissibility, strongly rejecting frameworks are more diverse. In particular, a strongly rejecting subframework $F|_S$ that is subset minimal (i.e., there is no S' with $S' \subsetneq S$ that strongly rejects) does not need to be admissible in F.

Example 4. *Consider an AF $F = (A, R)$ as shown in Figure 2(b). Two subset minimal strongly rejecting subframeworks for argument g under admissibility are $F|_{\{a,c,f\}}$ and $F|_{\{c,d,f\}}$. Neither $\{a, c, f\}$ nor $\{c, d, f\}$ is admissible in F. In this instance, $\{a, c\}$ or $\{d, f\}$ are not sufficient: in the first case one could add b, d, and g to $\{a, c\}$ resulting in $F|_{\{a,b,c,d,g\}}$ for which $\{b, d, g\}$ forms an admissible set. On the other hand $\{a, c, f\}$ strongly rejects g under adm: to defend g against c and f one requires both b and e, and to defend b against a one needs d. However, this means addition of all arguments, and g is not credulously accepted under admissibility in F.*

Nevertheless, admissible sets that are, in a sense, particularly strict are sufficient to make up a strongly rejecting subframework. More concretely, strongly admissible sets [4] that attack an argument constitute strongly rejecting subframeworks for this argument. An intuition can be seen in Example 2(a), in which both $\{a, c, h\}$ and $\{d, f, h\}$ are strongly admissible sets. We recall the formal definition of such sets next.

Definition 6 ([4, 13]). *Let $F = (A, R)$ be an AF. Set $S \subseteq A$ is strongly admissible in F if every $a \in S$ is defended by some $S' \subseteq S \setminus \{a\}$ which in turn is again strongly admissible.*

Intuitively, a set is strongly admissible if defense of that set can be "grounded" in unattacked arguments, with possibly using iterative defense. The next theorem formally states that strongly admissible sets attacking an argument each correspond to a strongly rejecting subframework of this argument.

Theorem 3. *Let $F = (A, R)$ be an AF and $a \in A$ an argument. If $S \subseteq A$ is strongly admissible in F and attacks a then $F|_S$ strongly rejects a under admissibility.*

Strongly admissible sets have been connected to particularly strong reasons for acceptance: strongly admissible sets are part of the grounded extension (with the unique subset maximal strongly admissible set being equal to the grounded extension), and strongly admissible sets are key components of a game-theoretic notion of showing acceptance of arguments (see, e.g., [13]). Thus, a strongly admissible set attacking an argument a represents a strong reason for rejection of argument a (under many semantics). Our notion of strongly rejecting frameworks capture such strongly admissible sets, in the sense of Theorem 3. That is, if a strongly admissible set exists attacking a, then there is also a corresponding strongly rejecting subframework for a under admissibility.

Relaxing the notion of strong admissibility is likely to prevent a direct correspondence as shown in Theorem 3. We show this formally by considering ideal sets [21]. An ideal set S is a set of arguments s.t. each argument is in all preferred extensions (skeptically accepted under preferred semantics), and S is, itself, also admissible. The unique subset maximal ideal set is the ideal extension of an AF, reflecting the same relation of strongly admissible sets and the grounded extension. Ideal sets (and the ideal extension) present a different viewpoint on what should be accepted in an AF when considering a rather cautious stance. However, the ideal extension E can, in general, be a superset of the grounded extension G of an AF.

Definition 7 ([21])**.** *Let $F = (A, R)$ be an AF. A set $S \subseteq A$ is ideal (in F) if $S \in adm(F)$ and $S \subseteq \bigcap_{E \in prf(F)} E$.*

As exemplified next, not every ideal set attacking an argument leads to a strongly rejecting subframework for that argument.

Example 5. *Consider an AF as shown in Figure 2(c). It holds that $\{d\}$ is an ideal set (ideal extension) of the AF. However, $F|_{\{d\}}$ does not strongly reject e, since e is credulously accepted under admissibility in $F|_{\{c,d,e\}}$. Thus, it is not the case that an ideal set S attacking e implies that $F|_S$ is strongly rejecting e. This can be the case if $F|_S$ leaves open ways of constructing an admissible set with e part of it.*

Intuitively, ideal sets are not “grounded” in unattacked arguments like strongly admissible sets, and, thus, do not directly constitute strongly rejecting subframeworks. This means that strongly admissible sets and ideal sets provide a kind of separation for admissible sets being necessarily strongly rejecting: if attacking an argument, the former leads to strong rejection while the latter might not.

Finally, in this section, we show that strongly rejecting frameworks $F|_S$ for admissibility can be characterized by non-existence of certain conflict-free sets in the original framework F. In the next proposition we omit the case with self-attacking arguments for readability.

Proposition 4. *Let $F = (A, R)$ be an AF, and $a \in A$ with $a \notin cred(F, adm)$ and $(a, a) \notin R$. It holds that $F|_S$ strongly rejects a under admissibility iff S attacks a and $\nexists S' \in cf(F)$ with $a \in S'$ and S' attacks each $x \in \{y \mid (y, z) \in R, y \in S, z \in S'\}$.*

Example 6. *Consider an AF $F = (A, R)$ with an illustration in Figure 2(d). That is, other than the two arguments a and b there are several further arguments c_i, each of them attacking b. The subframework $F|_{\{a\}}$ suffices to strongly reject b under admissibility, since each conflict-free set attacking $\{a\}$ (some subset of the c_i's) is always conflicting with b. As can be seen in this simple example (and formally by Proposition 4) for $\{a\}$ not inducing a strongly rejecting subframework, at least one c_i needs to be non-conflicting with b.*

Example 7. *For grounded semantics, strongly rejecting frameworks differ from admissibility. Consider an AF $F = (\{a, b, c, d\}, \{(a, b), (b, a), (a, c), (b, c), (c, d)\})$, i.e., four arguments with a and b mutually attacking each other and each of them attacking c, which in turn attacks d. While there are no strongly rejecting frameworks for d under adm and stb (since d is in stable extension $\{a, d\}$ for instance), the subframework $F|_{\{a,b,c\}}$ strongly rejects d under grounded semantics. Omission of any argument in $\{a, b, c\}$ leaves open the possibility of having d part of the grounded extension: without c the argument d is unattacked, leaving either a or b out the mutual attack is missing, and, e.g., $\{a, d\}$ is the grounded extension of $F|_{\{a,c,d\}}$.*

4 COMPLEXITY OF STRONGLY REJECTING FRAMEWORKS

We show the computational complexity of decision problems for finding small (w.r.t. cardinality) strongly rejecting subframeworks. We start with the complexity of verifying whether a subframework is strongly rejecting.

Proposition 5. *Verifying whether a given set of arguments in a given AF constitutes a subframework strongly rejecting a queried argument is coNP-complete for admissible and stable semantics.*

For the decision problem of deciding existence of a strongly rejecting framework of at most size k (which is a standard way of investigating complexity of optimization problems), we show that this problem is in Σ_2^P. As regards to hardness, we show both NP and coNP hardness.

Proposition 6. *Deciding whether there is a subframework strongly rejecting a queried argument under admissibility, with a number of arguments at most a given integer, is in Σ_2^P, and NP and coNP hard.*

That is, finding small strongly rejecting frameworks is plausibly more complex than checking credulous non-acceptance. Membership in Σ_2^P holds also for grounded and stable semantics. We conjecture that the preceding problem is, in fact, Σ_2^P-complete. Hardness for Σ_2^P is left for future work.

By Proposition 6, the minimum size (number of arguments) of a strongly rejecting framework for a queried argument can be found via querying a Σ_2^P oracle in a binary search. That is, computing a strongly rejecting subframework with a minimum number of arguments can be achieved by logarithmically many times solving the functional problem of computing a strongly rejecting subframework of at most a given size.

5 VARYING AND ENRICHING STRONG REASONS FOR REJECTION

Considering subframeworks induced by subsets of the arguments is one way of viewing strong reasons for rejections. In this section we argue that the underlying concept of strongly rejecting subframeworks, i.e., that of finding parts of the framework such that this part (and any super structure) rejects an argument, finds uses also in variants for different kinds of explanations. We exemplify two variants: having attacks inducing subframeworks, and relaxing semantical constraints.

Subframeworks induced by attacks Before, we specified subframeworks that are induced by subsets of arguments. We present now a variant that is based on attacks. To distinguish terminology from the notions above, we call a set $R' \subseteq R$ strongly attack-rejecting an argument $a \in A$, for a given AF $F = (A, R)$ under semantics σ iff it holds that for any $F' = (A, R'')$ with $R' \subseteq R'' \subseteq R$ we have $a \notin cred(F'', \sigma)$.

Example 8. *Considering the AF from Example 3 (Figure 2(a)), similarly as before we get that $\{(a, b), (c, g), (h, i)\}$) strongly attack-rejects i under adm (this AF is a direct counterpart to the strongly rejecting subframework from the example above). Omitting any of these attacks allows for an (attack) expansion that concludes credulous acceptance of i under admissibility.*

Differently to before, if an argument has multiple adjacent attacks (ingoing or outgoing) a strongly attack-rejecting set can focus on *the required attacks. E.g., in AF from Figure 2(b) a strongly attack-rejecting set for g under adm is* $\{(a,b),(c,g),(f,g),(d,e)\}$. *In this set, attacks from a to d, or vice versa, are not required to show strong attack-rejection of g. Moreover, the attack from d to e is now required, otherwise adding attacks* (d,a), (b,c), *and* (e,f) *would lead to credulous acceptance of g. That is, with attack-rejection, we highlight attacks needed for showing credulous non-acceptance.*

Relaxing argument wise defense We now consider a different form of rejection that is not based on the graph structure, but on the semantics. Recalling the criteria for admissibility, they state that a set $S \subseteq A$ is admissible if (a) S is conflict-free and (b) each member of S is defended by S. We relax the latter criteria in the following way, for an AF $F=(A,R)$. Let $X \subseteq A$ be a subset of arguments. We define a list of constraints that specify whether a set $S \subseteq A$ is admissible w.r.t. X (denoted by adm_X) as follows. A set $S \subseteq A$ that is conflict-free in F is admissible w.r.t. X iff all the following constraints are satisfied:

$$def_X = \{a \in S \text{ implies } a \text{ defended by } S \text{ in } F \mid a \in X\}.$$

That is, a conflict-free set S is adm_X if each $X \cap S$ is defended by S, while $S \setminus X$ need not be defended. For an AF $F=(A,R)$ we say that $X \subseteq A$ defense-rejects $a \in A$ if there is no set S with $a \in S$ s.t. S is adm_X. (In this case there is also no conflict-free S' with $a \in S'$ that is $adm_{X'}$ for $X \subseteq X'$; thus, supersets of X are already taken care of and need no explicit consideration).

Example 9. *Considering the simple AF from Example 3, it holds that* $X=\{b,g,i\}$ *defense-rejects i. To see this, there is no conflict-free set containing i that is* adm_X*, e.g., because b is never defended against the attack from a. Omitting any argument from X misses a reason for defense-rejection: e.g.,* $X'=\{b,i\}$ *implies that* $S=\{g,i\}$ *is* $adm_{X'}$*, since S is conflict-free, and i is defended by S (g is not defended by S, but this is not required by* $adm_{X'}$*). For the example AF in Figure 2(d), it suffices to consider* $X=\{b\}$*: no conflict-free set exists that defends b and contains b.*

Intuitively, this kind of relaxation of defense provides an alternative to structural views on explaining non-acceptability, by focusing on checking (non-)defense of a subset of arguments.

6 COMPUTING STRONG REJECTION VIA DECLARATIVE ASP ENCODINGS

In this section we show that answer set programming (ASP) can be used to compute strongly rejecting subframeworks, and, furthermore, also various other ways of showing strong rejection. We first recall background on ASP, and then show that declarative encodings of AF semantics can be utilized for finding strong reasons for rejection.

We recall basics of ASP next. A logic program π is a set of rules r of the form

$$\alpha_0 \leftarrow \alpha_1,\ldots,\alpha_m, not\ \alpha_{m+1},\ldots, not\ \alpha_n,\ \ 0 \le m \le n,$$

where each α_i is an atom and *not* is default negation; r is a *constraint* if α_0 is falsity ($\bot$, then omitted) and a *fact* if $n=0$. We also write r as $\alpha_0 \leftarrow B(r)$ or as $\alpha_0 \leftarrow B^+(r), not\ B^-(r)$, where $B^+(r)$ (positive body) is the set $\{\alpha_1,\ldots,\alpha_m\}$ and $B^-(r)$ (negative body) is the set $\{\alpha_{m+1},\ldots,\alpha_n\}$; furthermore, we let $B^\pm(r)=B^+(r)\cup B^-(r)$. We sometimes omit r from $B(r), B^\pm(r)$ when talking about a particular rule. Rules with variables stand for the set of their ground instances. The set of ground atoms of π is denoted by $\mathcal{A}$. Semantically, π induces a set of answer sets [31], which are Herbrand models (sets I of ground atoms) of π justified by the rules, in that I is a minimal model of $\pi^I=\{r \in \pi \mid I \models B(r)\}$ [24]. The set of answer sets of a program π is denoted as $AS(\pi)$. Common syntactic extensions are *choice rules* of the form $\{\alpha\} \leftarrow B$, which stands for the rules $\alpha \leftarrow B, not\ \alpha'$ and $\alpha' \leftarrow B, not\ \alpha$, where α' is a new atom.

Utilizing ASP for strong rejection To compute strong rejections we first show an encoding for verifying whether a set S constitutes a strong rejection, focusing initially on strongly rejecting subframeworks. We base our approach on well-known ASP encodings of AFs and AF semantics [23]. The graph structure of an AF $F=(A,R)$ is encoded by $\pi_F=\{\mathbf{arg}(a).\mid a \in A\}\cup\{\mathbf{att}(a,b).\mid (a,b)\in R\}$. We recall an encoding of admissibility in Listing 1 (several encodings π_σ for other semantics σ exist, as well). It has been shown that $E \in \sigma(F)$ iff there is an $I \in AS(\pi_\sigma \cup \pi_F)$ with $E=\{a \mid \mathbf{in}(a)\in I\}$ for $\sigma \in \{adm, stb\}$ and an AF $F=(A,R)$. We encode credulous acceptance of a queried argument $a \in A$ by constraint $\leftarrow not\ \mathbf{in}(a)$. In turn, an argument $a \in A$ is credulously accepted under σ if $\pi_\sigma \cup \pi_F \cup \{\leftarrow not\ \mathbf{in}(a).\}$ is satisfiable (the constraint removes all answer sets, respectively σ extensions, not containing a).

Listing 1. Encoding π_{adm} of admissible sets [23]

```
in(X) ← not out(X), arg(X).
out(X) ← not in(X), arg(X).
← in(X), in(Y), att(X,Y).
defeated(X) ← in(Y), att(Y,X).
undefended(X) ← att(Y,X), not defeated(Y).
← in(X), undefended(X).
```

For checking whether a set $S \subseteq A$ is a strongly rejecting framework for $a \in A$, we modify the encoding of arguments and attacks to allow for a non-deterministic guess of each superframework of $F|_S$ up to F. We do this by conditioning every attack on presence of both ingoing and outgoing argument, and introduce a guess for each argument $x \in A \setminus S$. To distinguish encodings, we use τ instead of π. We define $\tau_F=\{\mathbf{arg}(a) \leftarrow \mathbf{include}(a).\mid a \in A\}\cup\{\mathbf{att}(a,b)\leftarrow \mathbf{arg}(a),\mathbf{arg}(b).\mid (a,b)\in R\}$. Fixed inclusion of an argument $s \in S$ is then straightforward by specifying ASP fact $\mathbf{include}(s)$, and the remaining $x \in A \setminus S$ are guessed by the choice rule $\{\mathbf{include}(x)\}$. The resulting ASP encoding $\tau'=\tau_F\cup\{\mathbf{include}(s).\mid s \in S\}\cup\{\{\mathbf{include}(x)\}.\mid x \in A\setminus S\}$ has as its answer sets a correspondence to all frameworks $F|_{S'}$ with $S \subseteq S' \subseteq A$. Finally, $F|_S$ is strongly rejecting a iff $\tau'\cup\pi_{adm}\cup\{\leftarrow not\ \mathbf{in}(a).\}$ is unsatisfiable (mirroring our corresponding complexity result of Proposition 5).

In a similar way one can encode other reasons for rejection, e.g., for defense-rejection (see Section 5), one can condition the last rule of Listing 1 by **include** and in the same way have a fixed and guessed part (thus relaxing the constraint for specific arguments).

Towards a general approach of computing strong rejections, we make use of a recently proposed approach of abstraction in ASP [37], which, intuitively, gives a handle how to abstract certain ASP atoms in a systematic way, even when the atoms occur in rule bodies and not only as facts. Important for us is the following modification of an ASP program π by a set of atoms $\mathcal{L}\subseteq\mathcal{A}$. For every rule $r : \alpha \leftarrow B$ in π, we have

$$omit(r,\mathcal{L})=\begin{cases} r \text{ if } \mathcal{L}\cap B^\pm=\emptyset \wedge \alpha \notin \mathcal{L},\\ \{\alpha\}\leftarrow m_{\mathcal{L}}(B) \text{ if } \mathcal{L}\cap B^\pm \neq \emptyset \wedge \alpha \notin \mathcal{L}\cup\{\bot\},\\ \emptyset \text{ otherwise.}\end{cases}$$

where $m_{\mathcal{L}}(B)$ stands for $B^+(r) \setminus \mathcal{L}, not\ (B^-(r) \setminus \mathcal{L})$ ($m_{\mathcal{L}}$ is referred to as an omission abstraction mapping). In words, the rules are projected onto the non-omitted atoms and choice is introduced when an atom is omitted from a rule body, in order to make sure that the behavior of the original rule is preserved. The aim is that the atoms in $\mathcal{L}$ are to be "omitted" from rules, and the remaining atoms shall remain, while also ensuring that each original answer set of π can be projected onto some answer set of the constructed program $omit(\pi, \mathcal{L})$, thus achieving an *over-approximation*. The ground program $omit(\pi, \mathcal{L})$ is called an abstract program (reflecting omission of certain atoms). Due to the over-approximation, if $omit(\pi, \mathcal{L})$ is unsatisfiable, then π is unsatisfiable. That is, unsatisfiability is preserved. This property is used to find a cause of unsatisfiability in the following way.

Definition 8 ([37]). *A set $\mathcal{L} \subseteq \mathcal{A}$ of atoms is an* (answer set) blocker set *of π, if $omit(\pi, \mathcal{A} \setminus \mathcal{L})$ is unsatisfiable.*

In case $omit(\pi, \mathcal{A} \setminus \mathcal{L})$ is unsatisfiable, this means the atoms in $\mathcal{L}$ are blocking the occurrence of answer sets. No answer sets are possible as long as these atoms are present in the program.

Now, we can approach computation of strong rejections, and many variants, straightforwardly by considering blocker sets. We first show how blocker sets can be used for strongly rejecting subframeworks. When stating $\mathbf{include}(a)$ for each $a \in A$ as a fact, and defining omission on the list of atoms $\mathcal{L} = \{\mathbf{include}(x) \mid x \in A \setminus S\}$, the resulting abstract program $omit(\tau_F \cup \pi_{adm} \cup \{\mathbf{include}(b). \mid b \in A\} \cup \{\leftarrow not\ \mathbf{in}(a).\}, \mathcal{L})$ is unsatisfiable iff $F|_S$ is strongly rejecting a. Briefly put, for $x \in A \setminus S$, omission leads to a choice whether to include argument x (thus simulating the encoding proposed above).

As an example for computing different kinds of strongly rejecting reasons, consider omission of $\mathcal{L} = \{\mathbf{defeated}(x) \mid x \in A \setminus S\}$. Applying omission on the ground program via $omit(\pi_F \cup \pi_{adm} \cup \{\leftarrow not\ \mathbf{in}(a)\}, \mathcal{L})$, leads to removal of each rule with a $\mathbf{defeated}(x) \in \mathcal{L}$ in the head. Whenever x attacks an argument y, the omission modification induces a rule deriving $\{\mathbf{undefended}(y)\}$ instead of $\mathbf{undefended}(y)$ (last but one rule of Listing 1), effectively relaxing the prerequisite of defending y against x. This kind of omission is similar to defense-rejection (Section 5), but alleviates the need of defending arguments against certain attacking arguments, instead of not requiring to defend an argument at all.

Since ASP abstraction as presented here can simulate computation of strongly rejecting subframeworks (and other variants), as well as giving users a handle to find subsets of atoms of an ASP encoding that suffices to block satisfiability (block credulous acceptance) in a systematic way by specifying a list of atoms, we focus in the remainder of the technical part on computing small blocker sets.

Computing Minimum Blocker Sets We present an approach to find blocker sets of minimum cardinality, in order to find reasons for strong rejections that are small. That is, given a set of atoms $\mathcal{L}$ we aim to find an optimal $S^* \in arg\ min_{S \in U} |S|$ with $U = \{S \subseteq \mathcal{L} \mid omit(\pi, \mathcal{L} \setminus S)$ unsat.$\}$. As argued by our complexity results, and because computing minimum blocker sets needs to (at least) solve the problem of finding minimum sized strongly rejecting frameworks, finding small blocker sets is complex (more complexity results for blocker sets can be found in [37]). In brief, we encode the whole process of finding a blocker set of minimum cardinality via an encoding in disjunctive ASP with optimization statements (this ASP language is capable of solving optimization problems complete for Σ_2^P). In order to achieve that, the given ASP program π is turned into a meta ASP program, as shown in [29]. We modified construction of this meta program by incorporation of the omissions: freshly introduced atoms $\mathbf{omitted}(p(X))$ denote that $\mathbf{p}(X)$ is to be omitted. In other words, the input program (possibly encoding strongly rejecting subframeworks) is modified to include ASP atoms reflecting which atoms should be omitted, and then the whole program is modeled as a meta program. Such a meta program allows reasoning *on* the original ASP program. Critically, for us, we can reason on which atoms to omit (by specifying which $\mathbf{omitted}(p(X))$ to include) while preserving unsatisfiability. A saturation-based technique [29] is used to compute a maximum set of atoms that could be omitted while preserving unsatisfiability, which returns minimum blocker sets.

7 EXPERIMENTAL EVALUATION

We implemented computation of cardinality minimum blocker sets. The software, with further details, can be found at `http://www.kr.tuwien.ac.at/research/systems/abstraction/`. For evaluating our approach, we performed an experimental evaluation using AFs, and queried arguments, from the recent ICCMA'19 competition[2]. As ASP encodings we used the unmodified encodings for *adm* and *stb* of [23] (i.e., π_F and π_{adm} from above and π_{stb} from [23]). From the originally 326 instances of the competition (AF and queried argument), we took those directly for which the argument is not credulously accepted under admissibility. Whenever an argument is credulously accepted under admissibility, we computed new queries by selecting uniformly at random another argument, excluding those that were checked to be credulously accepted. This resulted in 211 instances (the remaining $326 - 211$ have all arguments credulously accepted under *adm*). The number of arguments in these AFs range from 7 to 10000 (median 475), and have a number of attacks from 9 to 1039471 (median 4374).

We performed computation of cardinality minimum blocker sets, with the restrictions that the blocker sets can only be composed of (i) attacks ($\mathbf{att}$), and (ii) $\mathbf{defeated}$ atoms, reflecting different types of blocker sets and different types of strong rejection. In addition, for admissibility we performed a straightforward preprocessing that computes the subgraph composed of all backwards reachable arguments from the queried argument, which is not applicable for stable semantics. We ran our ASP approach to compute a cardinality minimum blocker set, and set a timeout of 900 seconds and a memory limit of 8GB per individual instance. We used ASP solver clingo version 5.3.0 [28] and gringo 3.0.3 for meta programming [29].

We summarize performance evaluation in Figure 3, which shows a cactus plot, i.e., the number of instances (x-axis) that were solved within a certain time (y-axis) per type of abstractions. When considering $\mathbf{defeated}$ omissions, 85 instances and 95 instances were solved optimally, for *adm* and *stb* respectively (with a maximum number of arguments of 2130 and 4383, respectively). Our approach solved optimally about 50 instances for $\mathbf{att}$ omission. We hypothesize that the difference in running times for optimally solved instances is due to the induced search space: allowing for omission only on the $\mathbf{defeated}$ predicate represents a search space bounded by the number of arguments in an AF, while omissions on attacks implies a search space that is bounded by the number of attacks, which is itself bounded quadratically by the number of arguments. For the overall picture, while many instances were not solved optimally, we remark that up to 70 instances were solved with almost no time effort (Figure 3), and, furthermore, our approach also allows for non-optimal solutions to be returned as they are found (the ASP solver iteratively refines solutions until optimality is reached).

[2] `https://www.iccma2019.dmi.unipg.it/`

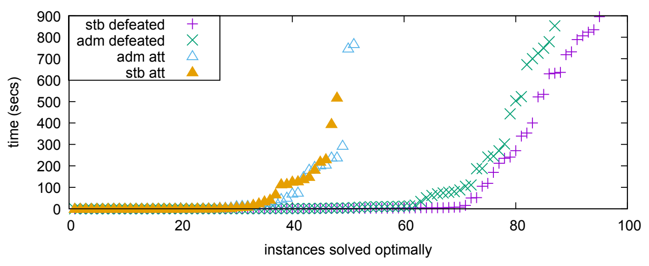

Figure 3. Cactus plot of optimally solved instances.

Regarding evaluation of blocker sets, we report the size of the optimal blocker sets that were found. In all cases, optimal blocker sets were very small: for $\mathbf{defeated}$ up to size 3 for admissibility and up to 6 for stability, and up to size 4 for all other parameters. This implies that the parts of the AF (or semantical criteria) necessary to find that the argument in question is to be rejected (credulously) are rather small, regarding the used ASP encoding and omission. We inspected several cases and found that the blocker sets show that only arguments up to a few steps away from the queried argument suffice to see non-acceptability. We note that in certain cases the ASP solver reduces the blocker set slightly by internal preprocessing. This in particular concerns unattacked arguments a, which do not occur in blocker sets (i.e., $\mathbf{defeated}(a)$ is not part of a blocker set).

8 DISCUSSION

Structural constraints and dependencies While (abstract) arguments in an AF do not have inherent dependencies other than attacks, such dependencies can be (implicitly) present when instantiating an AF from, e.g., a structured knowledge base. For instance, presence of an argument can imply presence of other arguments (e.g., when one is structurally a sub argument of the other). Our approach is versatile to accommodate such dependencies: since we specify what to omit via ASP atoms, dependencies between such omissions can be straightforwardly applied (via ASP rules). A recent study [41] on dependencies of instantiated arguments when applying modifications of the AF (as we do here by, e.g., shrinking the AF) shows that implications between presence of arguments can reflect important dependencies between arguments, which can be captured with our approach.

Related Work Explainability and (formal) argumentation naturally come together, and previous scientific works dealt with related questions. Within formal argumentation, probably the closest work is by [25] who look at (minimal) subframeworks, regarding arguments and attacks, s.t. an argument is not credulously accepted. In [14] games are studied that have a correspondence to admissible sets containing a queried argument that are subset minimal w.r.t. the admissible set and the arguments attacking this set. Abduction has been proposed for explainability in AFs: [36] defines explanations as modifications to an AF (adding or removing arguments) s.t. an argument can be labeled as in, out, or undecided, under labeling-based semantics [15]. In [9] so-called critical sets are studied which are subsets of the arguments in an AF s.t. each assignment of a complete labeling on the subset determines the labeling of arguments outside the subset. Recently, [6] proposed diagnoses when no argument is credulously (skeptically) accepted, under a semantics. For structured argumentation, within assumption-based argumentation, dispute trees were proposed to explain why a sentence should be concluded [17]. Formal approaches to argumentation, in varying forms, have also been employed to provide explanations for several domains, e.g., in [39, 35, 18]. In contrast to these works, we look at, e.g., subframeworks s.t. every expansion up to the original framework does not credulously accept a queried argument.

Regarding related work in the field of ASP, approaches to debugging explore explanations to unsatisfiable ASP programs [10, 33, 19, 30]. The main idea is to provide a reason of why an expected solution does not exist through occurring violations in the program and to steer the user towards the erroneous rules. In our setting, the program at hand is not considered to be buggy, and the aim is to get an understanding of which parts of the program are causing unsatisfiability. A survey on explanation approaches for ASP can be found in [26].

The idea of strong inconsistency was proposed recently by [11] for non-monotonic reasoning in general. In [12], strong inconsistency was used to derive explanations for credulous (skeptical) *acceptance* for logic programs. Our notions are inspired by theirs, where we directly tie strong rejection (strong inconsistency/explanations in their approach) to credulous non-acceptance under main semantic notions of AFs, and provide an in-depth analysis of these concepts by relating them to well-known concepts in argumentation, show complexity, provide an implementation, and an experimental evaluation. There is also a certain relation between strong inconsistency and strong equivalence; translating the corresponding result from [11] (Proposition 5.3) to our setting in terms of strong equivalence between AFs [34] amounts to the following statement: any subframework that is strongly equivalent (under semantics σ) to a subframework strongly rejecting argument a (under σ) is strongly rejecting a (under σ) itself. However, since strong equivalence between two AFs requires that the AFs are given over the same set of arguments [34], this results does not provide any additional insight, since for any two subframeworks $F = (A, R)$, $F' = (A', R')$ of an AF, $A = A'$ implies $F = F'$.

With the aim of studying general notions of equivalence within abstract argumentation, in [5] so-called C-restricted semantics were introduced for a core C of arguments in an AF. Their notion is related to our relaxation of defense via adm_X in the sense that C-restricted admissible semantics sanction conflict-free sets where only attacks from C need to be defended against, while our notion requires that a conflict-free set E of arguments needs only to defend its arguments that are within X. Relating C-restricted semantics to explanations remains as an interesting avenue for future work.

Conclusions In this paper we approached explanations of credulous non-acceptability under main argumentation semantics. Concretely, inspired by earlier work, we defined strongly rejecting subframeworks as subframeworks of AFs that provide sufficient reason for rejecting an argument credulously. Our findings are that such frameworks can be formally linked to particularly cautious game-theoretic notions of explanations, namely that of strong admissibility. Further, we exemplified that the underlying concept of strongly rejecting subframeworks can be varied to other notions, such as different types of subframeworks, or a relaxed notion of admissibility that looks at defense for only a subset of arguments. Such different notions can be tailored to which kind of explanation one seeks for in detailing credulous non-acceptance. While computational complexity of finding small reasons of rejection is high, as we show, we provided an implementation via ASP that can solve a reasonable number of AFs optimally, in the sense that smallest strong rejections are found. In our experiments of AF instances of ICCMA'19 all optimally solved instances resulted in very small blocker sets. This suggests that in many cases, despite the complexity barrier in the general case, small reasons of rejection can be found, and that, potentially, AF solvers can decide non-acceptance often in a very local manner, by investigating a small neighbourhood of a queried argument. More broadly, we view our work as a step towards argumentative explanation capabilities of AF systems at large.

ACKNOWLEDGEMENTS

This work was supported by the Austrian Science Fund (FWF): P30168-N31, Y698, and W1255-N23.

REFERENCES

[1] Katie Atkinson, Pietro Baroni, Massimiliano Giacomin, Anthony Hunter, Henry Prakken, Chris Reed, Guillermo Ricardo Simari, Matthias Thimm, and Serena Villata, 'Towards artificial argumentation', *AI Magazine*, **38**(3), 25–36, (2017).

[2] Pietro Baroni, Martin Caminada, and Massimiliano Giacomin, 'An introduction to argumentation semantics', *Knowledge Eng. Review*, **26**(4), 365–410, (2011).

[3] *Handbook of Formal Argumentation*, eds., Pietro Baroni, Dov Gabbay, Massimiliano Giacomin, and Leendert van der Torre, College Publications, 2018.

[4] Pietro Baroni and Massimiliano Giacomin, 'On principle-based evaluation of extension-based argumentation semantics', *Artif. Intell.*, **171**(10-15), 675–700, (2007).

[5] Ringo Baumann, Wolfgang Dvořák, Thomas Linsbichler, and Stefan Woltran, 'A general notion of equivalence for abstract argumentation', *Artif. Intell.*, **275**, 379–410, (2019).

[6] Ringo Baumann and Markus Ulbricht, 'If nothing is accepted - repairing argumentation frameworks', in *Proc. KR*, pp. 108–117. AAAI Press, (2018).

[7] Philippe Besnard and Anthony Hunter, *Elements of Argumentation*, MIT Press, 2008.

[8] Andrei Bondarenko, Phan Minh Dung, Robert A. Kowalski, and Francesca Toni, 'An abstract, argumentation-theoretic approach to default reasoning', *Artif. Intell.*, **93**, 63–101, (1997).

[9] Richard Booth, Martin Caminada, Paul E. Dunne, Mikolaj Podlaszewski, and Iyad Rahwan, 'Complexity properties of critical sets of arguments', in *Proc. COMMA*, volume 266, pp. 173–184. IOS Press, (2014).

[10] Martin Brain, Martin Gebser, Jörg Pührer, Torsten Schaub, Hans Tompits, and Stefan Woltran, 'Debugging ASP programs by means of ASP', in *Proc. LPNMR*, volume 4483 of *LNCS*, pp. 31–43. Springer, (2007).

[11] Gerhard Brewka, Matthias Thimm, and Markus Ulbricht, 'Strong inconsistency', *Artif. Intell.*, **267**, 78–117, (2019).

[12] Gerhard Brewka and Markus Ulbricht, 'Strong explanations for nonmonotonic reasoning', in *Description Logic, Theory Combination, and All That - Essays Dedicated to Franz Baader on the Occasion of His 60th Birthday*, eds., Carsten Lutz, Uli Sattler, Cesare Tinelli, Anni-Yasmin Turhan, and Frank Wolter, volume 11560 of *LNCS*, pp. 135–146. Springer, (2019).

[13] Martin Caminada and Paul E. Dunne, 'Strong admissibility revisited: Theory and applications', *Argument & Computation*, **10**(3), 277–300, (2019).

[14] Martin W. A. Caminada, Wolfgang Dvořák, and Srdjan Vesic, 'Preferred semantics as socratic discussion', *J. Log. Comput.*, **26**(4), 1257–1292, (2016).

[15] Martin W. A. Caminada and Dov M. Gabbay, 'A logical account of formal argumentation', *Studia Logica*, **93**(2-3), 109–145, (2009).

[16] Günther Charwat, Wolfgang Dvořák, Sarah A. Gaggl, Johannes P. Wallner, and Stefan Woltran, 'Methods for solving reasoning problems in abstract argumentation – A survey', *Artif. Intell.*, **220**, 28–63, (2015).

[17] Kristijonas Cyras, Xiuyi Fan, Claudia Schulz, and Francesca Toni, 'Assumption-based argumentation: Disputes, explanations, preferences', in *Handbook of Formal Argumentation*, eds., Pietro Baroni, Dov Gabbay, Massimiliano Giacomin, and Leendert van der Torre, chapter 7, 365–408, College Publications, (2018).

[18] Kristijonas Cyras, Dimitrios Letsios, Ruth Misener, and Francesca Toni, 'Argumentation for explainable scheduling', in *Proc. AAAI*, pp. 2752–2759. AAAI Press, (2019).

[19] Carmine Dodaro, Philip Gasteiger, Benjamin Musitsch, Francesco Ricca, and Kostyantyn Shchekotykhin, 'Interactive debugging of non-ground ASP programs', in *Proc. LPNMR*, volume 9345 of *LNCS*, pp. 279–293. Springer, (2015).

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

[21] Phan Minh Dung, Paolo Mancarella, and Francesca Toni, 'Computing ideal sceptical argumentation', *Artif. Intell.*, **171**(10-15), 642–674, (2007).

[22] Wolfgang Dvořák and Paul E. Dunne, 'Computational problems in formal argumentation and their complexity', in *Handbook of Formal Argumentation*, eds., Pietro Baroni, Dov Gabbay, Massimiliano Giacomin, and Leendert van der Torre, chapter 13, 631–688, College Publications, (2018).

[23] Uwe Egly, Sarah Alice Gaggl, and Stefan Woltran, 'Answer-set programming encodings for argumentation frameworks', *Argument & Computation*, **1**(2), 147–177, (2010).

[24] Wolfgang Faber, Nicola Leone, and Gerald Pfeifer, 'Recursive aggregates in disjunctive logic programs: Semantics and complexity', in *Proc. JELIA*, volume 3229 of *LNCS*, pp. 200–212. Springer, (2004).

[25] Xiuyi Fan and Francesca Toni, 'On explanations for non-acceptable arguments', in *Proc. TAFA, Revised Selected Papers*, volume 9524 of *LNCS*, pp. 112–127. Springer, (2015).

[26] Jorge Fandinno and Claudia Schulz, 'Answering the "why" in answer set programming - A survey of explanation approaches', *TPLP*, **19**(2), 114–203, (2019).

[27] Sarah Alice Gaggl, Thomas Linsbichler, Marco Maratea, and Stefan Woltran, 'Design and results of the second international competition on computational models of argumentation', *Artif. Intell.*, **279**, (2020).

[28] Martin Gebser, Roland Kaminski, Benjamin Kaufmann, Max Ostrowski, Torsten Schaub, and Philipp Wanko, 'Theory solving made easy with clingo 5', in *Technical Communications of ICLP*, volume 52 of *OASICS*, pp. 2:1–2:15. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, (2016).

[29] Martin Gebser, Roland Kaminski, and Torsten Schaub, 'Complex optimization in answer set programming', *TPLP*, **11**(4-5), 821–839, (2011).

[30] Martin Gebser, Jörg Pührer, Torsten Schaub, and Hans Tompits, 'A meta-programming technique for debugging answer-set programs.', in *Proc. AAAI*, pp. 448–453. AAAI Press, (2008).

[31] Michael Gelfond and Vladimir Lifschitz, 'Classical negation in logic programs and disjunctive databases', *New Generation Computing*, **9**(3), 365–385, (1991).

[32] Sanjay Modgil and Henry Prakken, 'A general account of argumentation with preferences', *Artif. Intell.*, **195**, 361–397, (2013).

[33] Johannes Oetsch, Jörg Pührer, and Hans Tompits, 'Catching the Ouroboros: On debugging non-ground answer-set programs', *TPLP*, **10**(4-6), 513–529, (2010).

[34] Emilia Oikarinen and Stefan Woltran, 'Characterizing strong equivalence for argumentation frameworks', *Artif. Intell.*, **175**(14-15), 1985–2009, (2011).

[35] Antonio Rago, Oana Cocarascu, and Francesca Toni, 'Argumentation-based recommendations: Fantastic explanations and how to find them', in *Proc. IJCAI*, pp. 1949–1955. ijcai.org, (2018).

[36] Chiaki Sakama, 'Abduction in argumentation frameworks', *Journal of Applied Non-Classical Logics*, **28**(2-3), 218–239, (2018).

[37] Zeynep G. Saribatur and Thomas Eiter, 'Omission-based abstraction for answer set programs', in *Proc. KR*, pp. 42–51. AAAI Press, (2018).

[38] Zeynep G. Saribatur, Peter Schüller, and Thomas Eiter, 'Abstraction for non-ground answer set programs', in *Proc. JELIA*, volume 11468 of *LNCS*, pp. 576–592. Springer, (2019).

[39] Dunja Seselja and Christian Straßer, 'Abstract argumentation and explanation applied to scientific debates', *Synthese*, **190**(12), 2195–2217, (2013).

[40] Matthias Thimm and Serena Villata, 'The first international competition on computational models of argumentation: Results and analysis', *Artif. Intell.*, **252**, 267–294, (2017).

[41] Johannes P. Wallner, 'Structural constraints for dynamic operators in abstract argumentation', *Argument & Computation*, (2019). Accepted manuscript available at `http://dx.doi.org/10.3233/AAC-190471`.