Skeptical Reasoning with Preferred Semantics in Abstract Argumentation without Computing Preferred Extensions

Matthias Thimm[1], **Federico Cerutti**[2] and **Mauro Vallati**[3]
[1]Institute for Web Science and Technologies, University of Koblenz-Landau, Germany
[2]Department of Information Engineering, University of Brescia, Italy
[3]School of Computing and Engineering, University of Huddersfield, United Kingdom
thimm@uni-koblenz.de, federico.cerutti@unibs.it, m.vallati@hud.ac.uk

Abstract

We address the problem of deciding skeptical acceptance wrt. preferred semantics of an argument in abstract argumentation frameworks, i. e., the problem of deciding whether an argument is contained in all maximally admissible sets, a.k.a. preferred extensions. State-of-the-art algorithms solve this problem with iterative calls to an external SAT-solver to determine preferred extensions. We provide a new characterisation of skeptical acceptance wrt. preferred semantics that does not involve the notion of a preferred extension. We then develop a new algorithm that also relies on iterative calls to an external SAT-solver but avoids the costly part of maximising admissible sets. We present the results of an experimental evaluation that shows that this new approach significantly outperforms the state of the art. We also apply similar ideas to develop a new algorithm for computing the ideal extension.

1 Introduction

Approaches to *formal argumentation* [Atkinson *et al.*, 2017; Baroni *et al.*, 2018] are knowledge representation formalisms that focus on the representation of arguments and their relationships. The most well-known approach is that of *abstract argumentation frameworks* [Dung, 1995] that model arguments as vertices in a directed graph, where a directed edge from an argument a to an argument b denotes an *attack* from a to b. By focusing only on the attack relations between arguments, they can seamlessly be used with a variety of approaches to structured argumentation such as ASPIC+ [Modgil and Prakken, 2014] and ABA [Toni, 2014]. The main purpose of abstract argumentation is to provide a machinery, or *semantics*, to identify justified and defeated arguments. One of the well-known semantics for abstract argumentation frameworks is the preferred semantics. A preferred extension is a maximal (wrt. set inclusion) admissible set, i. e., a set of arguments that is conflict-free and defends each of its members (we will provide definitions in Section 2). The computational problem of deciding whether an argument is contained in every preferred extension, i. e., whether it is *skeptically accepted*, is Π_2^{P}-complete [Dunne and Bench-Capon, 2002].

Interest in algorithms for solving reasoning problems in approaches to formal argumentation has recently increased, also fostered by the biennial International Competition of Computational Models of Arguments (ICCMA)[1], see also [Cerutti *et al.*, 2018] for a survey. Systems solving the skeptical acceptance problem wrt. preferred semantics often rely on SAT-solver technology [Biere *et al.*, 2009] or other paradigms such as answer set programming [Gebser *et al.*, 2012] to solve subproblems. A particular costly part in these algorithms is the maximisation step where, e. g., a series of SAT-solver calls are made to find ever larger admissible sets until a preferred extension is found. This approach is implemented in solvers such as Cegartix [Dvořák *et al.*, 2014], ArgSemSAT [Cerutti *et al.*, 2019], and μ-toksia [Niskanen and Järvisalo, 2020].

In this paper, we present a new characterisation of skeptical acceptance wrt. preferred semantics that does not rely on the notion of a preferred extension. We show that skeptical acceptance can be decided by considering only those admissible sets that are potentially conflicting with the acceptance status of the argument in question, and verifying that those admissible sets can be extended to include the argument, see Theorem 11 for the formal characterisation. Using this insight, we develop a new SAT-based algorithm that avoids the costly aspect of maximising admissible sets when deciding skeptical acceptance. Moreover, we apply similar concepts to the problem of determining the ideal extension, i. e. the unique maximal admissible set that is contained in every preferred extension. We compare the runtime performance of our new approaches with the best solvers from ICCMA19 [Bistarelli *et al.*, 2020] on the benchmark data of ICCMA19 and ICCMA17 [Gaggl *et al.*, 2020], and on an additional set of particularly hard problems. Our evaluation shows that our approaches significantly outperform the state of the art on both problems.

In summary, the contributions of this work are as follows:

1. We provide theoretical insights on the notion of skeptical acceptance wrt. preferred semantics and develop a new SAT-based algorithm (Section 4).
2. We generalise our theoretical results for the ideal extension and derive a corresponding algorithm (Section 5).
3. We conduct a thorough experimental analysis that shows that our approaches significantly outperform the former state of the art (Section 6).

[1]http://argumentationcompetition.org

Necessary background on abstract argumentation is given in Section 2 and related works are discussed in Section 3. We conclude with a discussion in Section 7.

Proofs of technical results are omitted due to space restrictions but can be found in an online appendix.[2]

2 Abstract argumentation

An *abstract argumentation framework* AF is a tuple $\mathsf{AF} = (\mathsf{A}, \mathsf{R})$ where A is a set of arguments and R is a relation $\mathsf{R} \subseteq \mathsf{A} \times \mathsf{A}$ [Dung, 1995]. For this work, we assume that A is always a finite set. For two arguments $a, b \in \mathsf{A}$ the relation $a\mathsf{R}b$ means that argument a attacks argument b. For a set $S \subseteq \mathsf{A}$ we define

$$S^+ = \{a \in \mathsf{A} \mid \exists b \in S, b\mathsf{R}a\}, S^- = \{a \in \mathsf{A} \mid \exists b \in S, a\mathsf{R}b\}.$$

We say that a set $S \subseteq \mathsf{A}$ is *conflict-free* if for all $a, b \in S$ it is not the case that $a\mathsf{R}b$, i. e., if $S \cap S^+ = \emptyset$. A set $S \subseteq \mathsf{A}$ *defends* an argument $b \in \mathsf{A}$ if for all a with $a\mathsf{R}b$ there is $c \in S$ with $c\mathsf{R}a$, i. e., if $\{b\}^- \subseteq S^+$. A conflict-free set S is called *admissible* if S defends all $a \in S$, i. e., if $S^- \subseteq S^+$.

Different semantics can be phrased by imposing further constraints on admissible sets [Dung, 1995; Dung *et al.*, 2007]. In this paper, we only consider preferred and ideal semantics. More precisely, an admissible set E

- is a *preferred extension* (PR) iff E is maximal (with respect to set inclusion),
- is an *ideal extension* (ID) iff $E \subseteq E'$ for all preferred extensions E' and E is maximal (with respect to set inclusion).

Note that the ideal extension always exists and is uniquely defined [Dung *et al.*, 2007, Thm. 2.1], while preferred extensions are also guaranteed to exist but are generally not uniquely defined [Dung, 1995]. For $\mathsf{AF} = (\mathsf{A}, \mathsf{R})$, we say an argument $a \in \mathsf{A}$ is *skeptically accepted wrt. preferred semantics* in AF if $a \in E$ for every preferred extension E. We denote by $\mathsf{Acc}(\mathsf{AF})$ the set of all arguments $a \in \mathsf{A}$ that are skeptically accepted wrt. preferred semantics in AF. The problem of deciding whether $a \in \mathsf{Acc}(\mathsf{AF})$ is Π_2^P-complete [Dunne and Bench-Capon, 2002]. Deciding whether an argument is contained in the ideal extension is Θ_2^P-complete under randomised reductions [Dunne, 2009].[3]

Example 1. Consider the abstract argumentation framework $\mathsf{AF} = (\mathsf{A}, \mathsf{R})$ depicted in Figure 1. The admissible sets of AF are

$$\emptyset, \{a\}, \{b\}, \{f\}, \{a, f\}, \{b, f\}, \{a, d, f\}, \{b, d, f\}$$

Here, $\{a, d, f\}$ and $\{b, d, f\}$ are preferred, making d and f skeptically accepted wrt. preferred semantics. The ideal extension of AF is $\{f\}$.

[2] http://mthimm.de/misc/proofs_ijcaj21_tcv.pdf

[3] Θ_2^P is the class of decision problems that can be solved in polynomial time by a deterministic Turing machine that can make a logarithmic number of calls to an NP oracle. A randomised reduction is, informally, a reduction that is valid "almost surely," see [Dunne and Bench-Capon, 2002] for technical details.

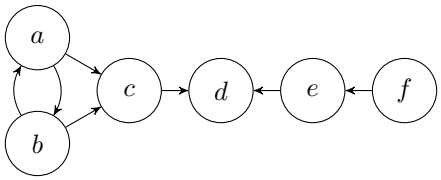

Figure 1: The argumentation framework AF from Example 1.

3 Related work

In the following, we briefly review the state of the art in algorithms and systems for deciding whether an argument is skeptically accepted wrt. preferred semantics.

In general, when designing algorithms for problems on the second level of the polynomial hierarchy, a popular way is to decompose them into sub-problems of *propositional satisfiability* (SAT) [Biere *et al.*, 2009], which in turn can be solved with calls to SAT solvers. A prominent family of approaches for this is the *Counter-Example-Guided Abstraction Refinement* (CEGAR) strategy [Clarke *et al.*, 2000]. This is the approach followed by the abstract argumentation solver Cegartix [Dvořák *et al.*, 2014] and subsequently in other approaches, including the winners for the preferred semantics tracks of the last two International Competitions of Computational Models of Argumentation (ICCMA): ArgSemSAT in 2017 [Cerutti *et al.*, 2019] and μ-toksia in 2019 [Niskanen and Järvisalo, 2020]. In the following, we focus on a detailed description of the CEGAR approach, due to its popularity and similarity to our own approach.

Algorithm 1 illustrates the CEGAR approach for deciding skeptical acceptance wrt. preferred semantics following [Dvořák *et al.*, 2014]. In order to describe the algorithm, we use some utility functions:

- $\mathsf{AdmArgAtt}(\mathsf{AF}, a)$ returns an admissible set S with $a \in S^+$ or FALSE if no such set exists.
- $\mathsf{AdmNotArg}(\mathsf{AF}, a, \{S_1, \ldots, S_n\})$ returns an admissible set S with $a \notin S$ and $S \not\subseteq S_1, \ldots, S \not\subseteq S_n$ or FALSE if no such set exists.
- $\mathsf{AdmSup}(\mathsf{AF}, S)$ returns an admissible set S' with $S \subsetneq S'$ or FALSE if no such set exists.

Observe that a single call to each of the above functions can be solved by a single call to a SAT-solver [Besnard and Doutre, 2004]. In the algorithm, it is first checked (lines 2–3) whether there is an admissible set that attacks the argument in question. Then we can already conclude that this argument cannot belong to every preferred extension. Line 4 then searches for an admissible set that does not contain the argument we are considering, and that is not a subset of the admissible sets we already considered before (stored in the variable $\mathcal{E}$ which has been initialised in line 1). Once such an admissible set is identified, the CEGAR procedure begins at line 5 where supersets of such an admissible set are searched for to identify a preferred extension. If then the argument in question does not belong to those supersets, we can conclude that it is not skeptically accepted.

Algorithm 1 Cegartix algorithm for skeptical acceptance wrt. preferred semantics [Dvořák *et al.*, 2014].

Input: $\mathsf{AF} = (\mathsf{A}, \mathsf{R}), a \in \mathsf{A}$
Output: YES iff $a \in \mathsf{Acc}(\mathsf{AF})$, NO otherwise

$\mathcal{E} := \emptyset$
if $\mathsf{AdmArgAtt}(\mathsf{AF}, a) \neq \text{FALSE}$ **then**
  return NO
while $S := \mathsf{AdmNotArg}(\mathsf{AF}, a, \mathcal{E}) \neq \text{FALSE}$ **do**
  while $S' := \mathsf{AdmSup}(\mathsf{AF}, S) \neq \text{FALSE}$ **do**
    $S := S'$
  if $a \notin S$ **then**
    return NO
  else
    $\mathcal{E} := \mathcal{E} \cup \{S\}$
return YES

4 Skeptical acceptance revisited

We start our investigation with some auxiliary notions and general observations on skeptical acceptance wrt. preferred semantics, many of which have been used implicitly in other works. Let $\mathsf{AF} = (\mathsf{A}, \mathsf{R})$ be fixed throughout the rest of this paper.

A core property of skeptical acceptance for an argument $a \in \mathsf{A}$ is that a is "compatible" with every admissible set $S \subseteq \mathsf{A}$. Here, the notion of "compatibility" carries two constraints. First, it is necessary that a is contained in some admissible set. Second, every admissible set can be "extended" to include a. We make these aspects more concrete in the following.

Let us first define the *admissible core* as the set of all arguments that are in at least one admissible set.

Definition 2. The *admissible core* $\mathsf{AC}(\mathsf{AF})$ of AF is $\mathsf{AC}(\mathsf{AF}) = \{a \in \mathsf{A} \mid \text{ there is an admissible } S \text{ with } a \in S\}$.

As preferred extensions are maximal admissible sets, it is clear that arguments not contained in any admissible set cannot be skeptically accepted wrt. preferred semantics. The following observation is therefore given without proof.

Proposition 3. $\mathsf{Acc}(\mathsf{AF}) \subseteq \mathsf{AC}(\mathsf{AF})$.

Now, the second aspect of skeptical acceptance wrt. preferred semantics is that admissible sets can be "extended" to include the argument in question. One observation regarding this—which is already exploited in Algorithm 1—is that arguments attacked by some admissible set cannot be skeptically accepted. The following notion of attacks between sets of arguments will come in handy.

Definition 4. A set $S \subseteq \mathsf{A}$ *attacks* an argument $a \in \mathsf{A}$ in AF, denoted as $S\mathsf{R}a$, if $a \in S^+$. A set $S_1 \subseteq \mathsf{A}$ *attacks* a set $S_2 \subseteq \mathsf{A}$ in AF if $S_1^+ \cap S_2 \neq \emptyset$.

We define the *preferred super-core* as the set of arguments which are not attacked by an admissible set as follows.

Definition 5. The *preferred super-core* $\mathsf{PSC}(\mathsf{AF})$ of AF is

$$\mathsf{PSC}(\mathsf{AF}) = \{a \in \mathsf{A} \mid \text{ there is no admissible } S \text{ with } S\mathsf{R}a\}$$

As every admissible set is a subset of some preferred extension, an admissible set S with $S\mathsf{R}a$ is included in a preferred extension that does not contain a (otherwise this extension would not be conflict-free). Therefore, the following observation is also clear.

Proposition 6. $\mathsf{Acc}(\mathsf{AF}) \subseteq \mathsf{PSC}(\mathsf{AF})$.

Together with Proposition 3 we obtain the following corollary.

Corollary 7. $\mathsf{Acc}(\mathsf{AF}) \subseteq \mathsf{AC}(\mathsf{AF}) \cap \mathsf{PSC}(\mathsf{AF})$.

Now we make precise what it means that every admissible set can be extended to include a skeptically accepted argument.

Proposition 8. *For $a \in \mathsf{Acc}(\mathsf{AF})$ and an admissible set S there is an admissible set S' with $S \subseteq S'$ and $a \in S'$.*

Putting things together, we can actually *characterise* skeptical acceptance wrt. preferred semantics as follows.

Theorem 9. *$a \in \mathsf{Acc}(\mathsf{AF})$ if and only if*

1. *$a \in \mathsf{AC}(\mathsf{AF})$ and*
2. *for every admissible set S there is an admissible set S' with $S \subseteq S'$ and $a \in S'$.*

The interesting aspect of the above theorem is that it characterises skeptical acceptance wrt. preferred semantics without the notion of a preferred extension. It shows that skeptical acceptance wrt. preferred semantics can be fully characterised by admissibility and subset relationships of admissible sets. Therefore, the computationally expensive part of maximising an admissible set to yield a preferred extension can be avoided in algorithmic solutions. However, one caveat of Theorem 9 in that regard is that possibly many admissible sets have to be checked to verify whether an argument is skeptically accepted. Using the following observation—which is a straightforward generalisation of Dung's Fundamental Lemma [Dung, 1995]—we can strengthen our above characterisation even further.

Lemma 10. *For admissible sets S and S', if not $S\mathsf{R}S'$ and not $S'\mathsf{R}S$ then $S \cup S'$ is admissible.*

The above lemma suggests that we do not have to consider all admissible sets in step 2 of Theorem 9 but only those that provide some conflict with admissible sets containing the argument under consideration. Taking this into account, our final characterisation of skeptical acceptance under preferred semantics is as follows.

Theorem 11. *$a \in \mathsf{Acc}(\mathsf{AF})$ if and only if*

1. *$a \in \mathsf{AC}(\mathsf{AF})$ and*
2. *for every admissible set S with $a \in S$ and every admissible set S' with $S'\mathsf{R}S$, there is an admissible set S'' with $S' \cup \{a\} \subseteq S''$.*

Theorem 11 states that we can decide skeptical acceptance wrt. preferred semantics by considering only those admissible sets that attack an admissible set containing the argument in question.

Algorithm 2, denoted CDAS (*Conflict-Driven Admissibility Search*), exploits Theorem 11 to determine skeptical acceptance wrt. preferred semantics. For that, we will use some further utility functions:

Algorithm 2 CDAS algorithm for skeptical acceptance wrt. preferred semantics

Input: $\mathsf{AF} = (\mathsf{A}, \mathsf{R}), a \in \mathsf{A}$
Output: YES iff $a \in \mathsf{Acc}(\mathsf{AF})$, NO otherwise

```
1:  S := AdmExt(AF, {a})
2:  if S = FALSE then
3:      return NO
4:  E := ∅
5:  while TRUE do
6:      S' := AdmExtAtt(AF, {a}, E)
7:      if S' = FALSE then
8:          return YES
9:      S'' := AdmExt(AF, S' ∪ {a})
10:     if S'' = FALSE then
11:         return NO
12:     E := E ∪ {S''}
```

- $\mathsf{AdmExt}(\mathsf{AF}, S)$ returns an admissible set S' with $S \subseteq S'$ or FALSE if no such set exists.
- $\mathsf{AdmExtAtt}(\mathsf{AF}, S, \{S_1, \ldots, S_n\})$ returns an admissible set S' s.t. there is an admissible set S'' with
  1. $S \subseteq S''$,
  2. $S' \mathsf{R} S''$, and
  3. $S' \not\subseteq S_i$ for $i = 1, \ldots, n$.

  If there is no such S' then FALSE is returned.

Observe that a single call to each of the above functions can be solved by a single call to a SAT-solver.[4]

Lines 1–3 of Algorithm 2 verify condition 1 of Theorem 11: if the argument a is not contained in an admissible set, the algorithm terminates with answer NO. Lines 4–12 verify condition 2 of Theorem 11. Here, the set $\mathcal{E}$ keeps track of admissible sets already containing the argument a. Line 6 looks for an admissible set S' that attacks some admissible set containing a. Those sets which are subsets of previously considered sets in $\mathcal{E}$ are ignored. If such a set S' cannot be found, we have shown that a must be skeptically accepted and the algorithm terminates with YES (lines 7–8). Otherwise, S' should be extended to include a, yielding an admissible set S''. If no such set can be found, the algorithm terminates with NO (lines 10–12). Otherwise, the set S'' is stored in $\mathcal{E}$ to avoid considering subsets of S'' later (line 12).

Theorem 12. *Algorithm 2 is sound and complete.*

We will see in Section 6 that our approach is feasible in practice and significantly outperforms the former state of the art.

5 Ideal semantics revisited

Note that Theorem 11 characterises when a single argument is skeptically accepted wrt. preferred semantics. This characterisation is actually a special case of the following more general result about the set $\mathsf{Acc}(\mathsf{AF})$ of all skeptically accepted arguments wrt. preferred semantics.

[4] In particular, note that AdmExtAtt basically consists of guessing two sets S' and S'' satisfying the listed conditions.

Theorem 13. *Let $T \subseteq \mathsf{A}$ satisfy the following conditions:*

1. *there is an admissible set S with $T \subseteq S$,*
2. *for every admissible set S with $T \subseteq S$ and every admissible set S' with $S' \mathsf{R} S$ there is an admissible set S'' with $T \cup S' \subseteq S''$, and*
3. *there is no T' with $T \subsetneq T'$ satisfying conditions (1) and (2).*

Then T exists, is uniquely determined, and $T = \mathsf{Acc}(\mathsf{AF})$.

In other words, $\mathsf{Acc}(\mathsf{AF})$ is the maximal set of arguments that is contained in at least one admissible set and is "compatible" with every admissible set.

Recall now that the ideal extension E_{id} of an abstract argumentation framework $\mathsf{AF} = (\mathsf{A}, \mathsf{R})$ is the unique maximal admissible set contained in $\mathsf{Acc}(\mathsf{AF})$. Then Theorem 13 can also be slightly adapted to characterise the ideal extension instead.

Theorem 14. *Let $T \subseteq \mathsf{A}$ satisfy the following conditions:*

1. *T is admissible,*
2. *for every admissible set S with $T \subseteq S$ and every admissible set S' with $S' \mathsf{R} S$ there is an admissible set S'' with $T \cup S' \subseteq S''$, and*
3. *there is no T' with $T \subsetneq T'$ satisfying conditions (1) and (2).*

Then T is the ideal extension.

In other words, the ideal extension is the maximal admissible set that is "compatible" with every other admissible set.

Another characterisation of the ideal extension, which yields even better performance when exploited in algorithms,[5] can be found in [Dung *et al.*, 2007]. Using our notation, this characterisation is as follows.

Theorem 15 (Theorem 3.3 in [Dung *et al.*, 2007])**.** *The ideal extension E_{id} of AF is the largest (wrt. set inclusion) admissible set in $\mathsf{PSC}(\mathsf{AF})$.*

The interesting aspect of the above theorem—which seems to have been overlooked for algorithms until now—is that ideal semantics, although defined based on skeptical acceptance wrt. preferred semantics, does not rely on that notion. Only one aspect of skeptically accepted arguments wrt. preferred semantics is needed to define the ideal extension, namely the property of not being attacked by an admissible set. This characterisation can be exploited by algorithms, particularly because of the following observation.

Proposition 16. *Given $\mathsf{PSC}(\mathsf{AF})$, the ideal extension E_{id} can be computed in polynomial time.*

Algorithm 3 outlines our algorithm for computing the ideal extension, denoted CDIS (*Conflict-Driven Ideal Search*). For that, we will use the following utility function:

- $\mathsf{AdmAttExt}'(\mathsf{AF}, P)$ returns an admissible set S that attacks P or FALSE if no such set exists.

Note also that $\mathsf{AdmAttExt}'(\mathsf{AF}, S)$ can be implemented by a single call to a SAT solver.

[5] This insight has been gained from preliminary experiments we do not report here.

Algorithm 3 CDIS algorithms for computing the ideal extension

Input: $\mathsf{AF} = (\mathsf{A}, \mathsf{R})$
Output: The ideal extension E_{id} of AF
1: $P := \mathsf{A}$
2: **while** TRUE **do**
3: $S := \mathsf{AdmAttExt}'(\mathsf{AF}, P)$
4: **if** $S =$ FALSE **then**
5: **break**
6: $P := P \setminus S^+$
7: $P := P \setminus P^+$
8: **while** $P \neq P \setminus (\, P^- \setminus P^+ \,)^+$ **do**
9: $P := P \setminus (\, P^- \setminus P^+ \,)^+$
10: **return** P

In lines 1–6, the algorithm computes the preferred supercore P of AF, i. e., the set of arguments which are not attacked by any admissible set. It iteratively looks for an admissible set S that attacks at least one argument in the candidate set P (line 3) and then removes all arguments attacked by S from the candidate set P (line 6). Afterwards (see also the proof of Proposition 16), it removes all arguments attacked by another argument in P (line 7) as those cannot be defended. Then it iteratively removes arguments from P that are attacked but not defended within P. In particular, note that $(P^- \setminus P^+)^+$ is the set of all arguments attacked by some argument a that attacks some argument in P and is not attacked back by any argument in P. The arguments that remain form the ideal extension.

Theorem 17. *Algorithm 3 is sound and complete.*

Algorithm 3 differs from previous approaches to compute the ideal extension but is complementary to the algorithm from [Dunne, 2009], which is also implemented in, e. g., Cegartix [Dvořák *et al.*, 2014]. Roughly speaking, that algorithm first computes all arguments that are contained in an admissible set and then iteratively removes the arguments that are attacked within that set. Conversely, our algorithm first computes all arguments that are attacked by an admissible set and then removes the undefended arguments until only the ideal extension remains. Under the assumption that there are fewer arguments not attacked by any admissible set than there are arguments contained in an admissible set, the advantage of our algorithm is that the handled arguments are fewer and it converges sooner to the ideal extension. We will provide experimental evidence for that in the next section.

6 Experiments

The goal of our experimental evaluation is to show that our algorithms solve the decision problem of skeptical acceptance wrt. preferred semantics (also denoted as DS-PR in the ICCMA competitions) and the computation of the ideal extension (SE-ID) faster than the state-of-the-art algorithms.

6.1 Experimental setup

We evaluate the runtime performance on the same benchmark data as used in the previous competitions ICCMA17 and ICCMA19, and on an additional benchmark set containing particularly hard problems. In more detail:

1. For DS-PR we used the benchmark data instances A2–A5 from ICCMA17 (350 instances) and the whole benchmark data set from ICCMA19 (326 instances), both with the prescribed query arguments.
2. For SE-ID we used the benchmark data instances D1–D5 from ICCMA17 (350 instances) and the whole benchmark data set from ICCMA19 (326 instances).
3. For both DS-PR and SE-ID, we additionally generated 252 random graphs (WS-hard) of the Watts-Strogatz graph model [Watts and Strogatz, 1998] using the AFBenchGen2 suite [Cerutti *et al.*, 2016].[6] These graphs have a number of arguments between 300 and 600 and were generated using parameter values between 10 and 40 for `-WS_baseDegree`, between 0.2 and 0.6 for `-WS_beta`, and between 0.2 and 0.6 for `-BA_WS_probCycles`, see [Cerutti *et al.*, 2016] for a detailed description of these parameters. Query arguments for DS-PR have been selected for each generated graph uniformly at random.

We implemented our algorithms CDAS (Algorithm 2) and CDIS (Algorithm 3) in C++ using standard data structures and used Glucose 4.1 in its non-parallel version [Audemard and Simon, 2018] for all SAT calls. The resulting system has been called *Fudge*.[7]

For both DS-PR and SE-ID, we compared Fudge with the top three solvers of ICCMA19[8] for the DS-PR track: μ-toksia [Niskanen and Järvisalo, 2020] (version 2020.03.12), CoQuiAAS [Lagniez *et al.*, 2019] (version 3.0), and ASPARTIX [Dvořák *et al.*, 2020] (version V19). Note that μ-toksia was the best solver for both DS-PR and SE-ID in ICCMA19.

The experiments were conducted on a dedicated server with an Intel Xeon CPU (2.9 GHz) and 128 GB RAM running Ubuntu 20.04.1. We set a 600s CPU-time cutoff time, which is the same used for the ICCMA competitions. Results are presented in terms of cumulative runtime over solved instances (RT), number of instances not solved within the cutoff time (TO), and P10 (Penalised average runtime 10). P10 is a metric usually exploited for algorithm configuration, where the average runtime is calculated by considering runs that did not solve the problem as ten times the cutoff time.

6.2 Results

The results of our experimental analysis are summarised in Table 1. Our solver Fudge is consistently delivering the best performance, according to the considered metrics, on the benchmark sets. In some cases (such as for DS-PR on WS-hard), the cumulative runtime is higher than those of other solvers, but the number of solved instances is also significantly larger. Also note that, e. g., for DS-PR on

[6] These graphs seem to be particularly hard for both problems DS-PR and SE-ID. For example, the benchmark set A5 from ICCMA17 contained 20 such instances and all of them resulted in a timeout for, e. g., μ-toksia on DS-PR.

[7] The source code is available at http://taas.tweetyproject.org.

[8] http://argumentationcompetition.org/2019/results

	DS-PR								
	ICCMA17 (350)			ICCMA19 (326)			WS-hard (252)		
	RT	TO	P10	RT	TO	P10	RT	TO	P10
ASPARTIX	6765.9	55	962.2	1327.0	**0**	4.1	3905.8	111	2658.4
μ-toksia	5976.2	46	805.6	194.7	**0**	0.6	2782.6	103	2463.4
CoQuiAAS	5375.7	85	1472.5	1014.0	**0**	3.1	2415.8	124	2962.0
Fudge	**3550.2**	**21**	**370.1**	**101.5**	**0**	**0.3**	**5272.4**	**71**	**1711.4**

	SE-ID								
	ICCMA17 (350)			ICCMA19 (326)			WS-hard (252)		
	RT	TO	P10	RT	TO	P10	RT	TO	P10
ASPARTIX	9519.5	55	970.1	1311.3	3	59.2	5278.5	108	2592.4
μ-toksia	5593.5	18	324.6	607.3	**0**	1.9	2119.9	104	2484.6
CoQuiAAS	6200.5	65	1132.0	1501.2	1	23.0	2426.0	124	2962.0
Fudge	**5637.5**	**11**	**204.7**	**240.6**	**0**	**0.7**	**3748.8**	**98**	**2348.2**

Table 1: Performance of the considered systems on the selected benchmark sets when dealing with the DS-PR (top) and SE-ID (bottom) problems, numbers between brackets refer to the number of instances of the set. RT, TO, P10 indicates, respectively, cumulative runtime over all solved instances, number of timeouts, and Penalised Average Runtime 10 score. Boldface entries highlight best performance in the respective column; for column RT, the best performance among systems with the minimal number of timeouts is highlighted.

the ICCMA17 data set, Fudge was able to solve considerably more instances in less time than the other solvers. A Wilcoxon signed-rank test ($p < 0.05$) confirms that the differences between the performance of Fudge and the performance of the other considered solvers are statistically significant.

Figure 2 shows a cactus plot on how the number of solved instances is affected by the available CPU-time, when dealing with DS-PR and SE-ID, accumulated over all instances from all three benchmark sets. Finally, Figure 3 presents a comparison between the runtimes of μ-toksia, winner of the DS-PR track of ICCMA19, and Fudge for DS-PR. The dots above (below) the diagonal represent instances where Fudge was faster (slower) than μ-toksia.

It is worth mentioning that both Fudge and the runner-up μ-toksia used the exact same SAT-solver for solving subproblems, i. e., Glucose 4.1 in its non-parallel version [Audemard and Simon, 2018] (CoQuiAAS and ASPARTIX rely on different problem solving paradigms). Furthermore, the encodings used in utility functions of the form Adm* in both μ-toksia and Fudge are derived from the encodings of [Besnard and Doutre, 2004] and quite similar. Therefore, the difference in the performances of these two systems can only be explained by the general strategy used for decomposing the argumentation problems into a series of satisfiability checks. An analysis of the number of SAT calls each solver makes during a single instance reveals that Fudge only makes about 1/3 as many SAT calls as μ-toksia does (median value over all instances that were solved by both solvers).

7 Summary and conclusion

We presented new insights into the concept of skeptical acceptance wrt. preferred semantics (and to some extent also into the concept of the ideal extension) by showing that skeptical acceptance can be characterised without relying on the notion of a preferred extension. On this basis, we developed new algorithms that have been shown to clearly outperform the previous state of the art.

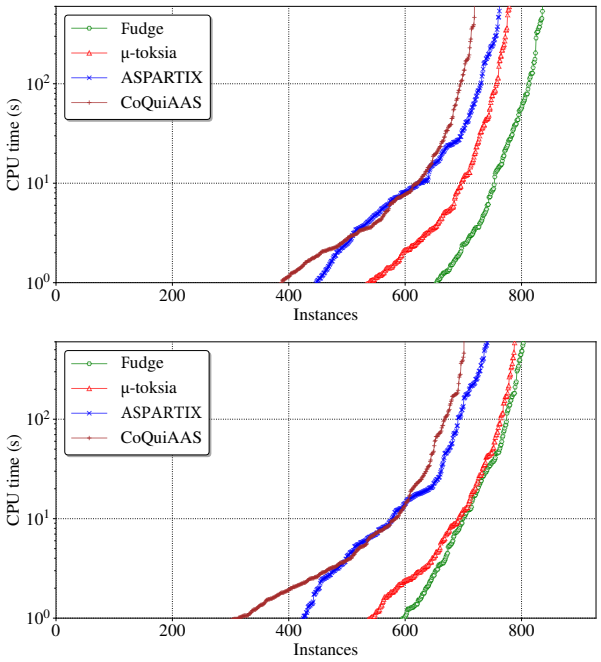

Figure 2: Number of solved instances according to the available CPU-time, when dealing with DS-PR (top) and SE-ID (bottom). The data is accumulated over all instances from all tested benchmark sets (928 instances for both problems).

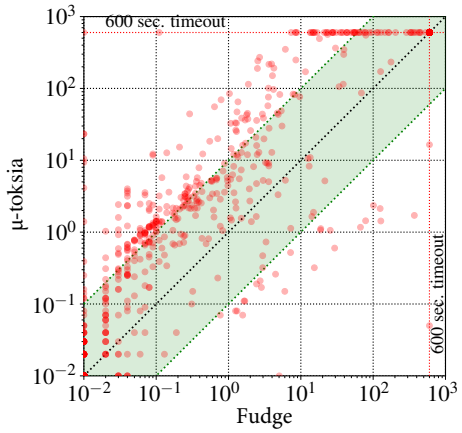

Figure 3: Scatter plot comparing the runtime of μ-toksia (y-axis) and Fudge (x-axis) when dealing with DS-PR over all considered instances (928). Instances outside the green area indicate runtime differences up to an order of magnitude.

For future work we plan, among others, to extend our work to deal with *dynamic problems* as well, i. e., problems where the acceptance status of an argument has to be recomputed after addition or deletion of attacks, cf. the dynamics track of ICCMA 2021.

Acknowledgements

Mauro Vallati was supported by a UKRI Future Leaders Fellowship [grant number MR/T041196/1].

References

[Atkinson *et al.*, 2017] K. Atkinson, P. Baroni, M. Giacomin, A. Hunter, H. Prakken, C. Reed, G. R. Simari, M. Thimm, and S. Villata. Toward artificial argumentation. *AI Magazine*, 38(3):25–36, 2017.

[Audemard and Simon, 2018] G. Audemard and L. Simon. On the Glucose SAT solver. *International Journal on Artificial Intelligence Tools*, 27(01), 2018.

[Baroni *et al.*, 2018] P. Baroni, D. M. Gabbay, M. Giacomin, and L. van der Torre, editors. *Handbook of Formal Argumentation*. College Publications, 2018.

[Besnard and Doutre, 2004] P. Besnard and S. Doutre. Checking the acceptability of a set of arguments. In *Proceedings of the 10th International Workshop on Non-Monotonic Reasoning (NMR'04)*, 2004.

[Biere *et al.*, 2009] A. Biere, M. Heule, H. van Maaren, and T. Walsh, editors. *Handbook of Satisfiability*. IOS Press, 2009.

[Bistarelli *et al.*, 2020] S. Bistarelli, L. Kotthoff, F. Santini, and C. Taticchi. A first overview of ICCMA'19. In *Proceedings of the Workshop on Advances in Argumentation in Artificial Intelligence 2020 co-located with the 19th International Conference of the Italian Association for Artificial Intelligence (AIxIA 2020)*, pages 90–102, 2020.

[Cerutti *et al.*, 2016] F. Cerutti, M. Giacomin, and M. Vallati. Generating structured argumentation frameworks: AFBenchGen2. In *Proceedings of the 6th International Conference of Computational Models of Argument (COMMA'16)*, pages 467–468, 2016.

[Cerutti *et al.*, 2018] F. Cerutti, S. A. Gaggl, M. Thimm, and J. P. Wallner. Foundations of implementations for formal argumentation. In *Handbook of Formal Argumentation*, chapter 15. College Publications, 2018.

[Cerutti *et al.*, 2019] F. Cerutti, M. Giacomin, and M. Vallati. How we designed winning algorithms for abstract argumentation and which insight we attained. *Artificial Intelligence*, 276:1 – 40, 2019.

[Clarke *et al.*, 2000] E. Clarke, O. Grumberg, S. Jha, Y. Lu, and H. Veith. Counterexample-guided abstraction refinement. In *Computer Aided Verification*, pages 154–169, Berlin, Heidelberg, 2000. Springer Berlin Heidelberg.

[Dung *et al.*, 2007] P. M. Dung, P. Mancarella, and F. Toni. Computing ideal sceptical argumentation. *Artificial Intelligence*, 171(10):642–674, 2007.

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

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

[Dunne, 2009] P. E. Dunne. The computational complexity of ideal semantics. *Artificial Intelligence*, 173(18):1559–1591, 2009.

[Dvořák *et al.*, 2020] W. Dvořák, A. Rapberger, J. P. Wallner, and S. Woltran. ASPARTIX-V19: An answer-set programming based system for abstract argumentation. In *Proceedings of the 11th International Symposium on Foundations of Information and Knowledge Systems (FoIKS'11)*, pages 79–89, 2020.

[Dvořák *et al.*, 2014] W. Dvořák, M. Järvisalo, J. P. Wallner, and S. Woltran. Complexity-sensitive decision procedures for abstract argumentation. *Artificial Intelligence*, 206:53–78, 2014.

[Gaggl *et al.*, 2020] S. A. Gaggl, T. Linsbichler, M. Maratea, and S. Woltran. Design and results of the second international competition on computational models of argumentation. *Artificial Intelligence*, 279:103193, 2020.

[Gebser *et al.*, 2012] M. Gebser, R. Kaminski, B. Kaufmann, and T. Schaub. *Answer Set Solving in Practice*. Morgan & Claypool Publishers, 2012.

[Lagniez *et al.*, 2019] J.-M. Lagniez, E. Lonca, and J.-G. Mailly. CoQuiAAS v3.0 – ICCMA 2019 Solver Description. In *System descriptions of the Third International Competition on Computational Models of Argumentation (ICCMA'19)*, 2019.

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

[Niskanen and Järvisalo, 2020] A. Niskanen and M. Järvisalo. μ-toksia: An efficient abstract argumentation reasoner. In *Proceedings of the 17th International Conference on Principles of Knowledge Representation and Reasoning (KR 2020)*, 2020.

[Toni, 2014] F. Toni. A tutorial on assumption-based argumentation. *Argument & Computation*, 5(1):89–117, 2014.

[Watts and Strogatz, 1998] D. J. Watts and S. H. Strogatz. Collective dynamics of 'small-world' networks. *Nature*, 393(6684):440–442, 1998.