Proceedings of the 20th International Conference on Principles of Knowledge Representation and Reasoning
Main Track

Forgetting Aspects in Assumption-Based Argumentation

Matti Berthold[1] , **Anna Rapberger**[2] and **Markus Ulbricht**[1]
[1]ScaDS.AI, Dresden/Leipzig, Universität Leipzig, Germany
[2]TU Wien, Austria
{berthold, mulbricht}@informatik.uni-leizpig.de, anna.rapberger@tuwien.ac.at

Abstract

We address the issue of forgetting in assumption-based argumentation (ABA). Forgetting is driven by the goal to remove certain elements from a knowledge base, while preserving the structure of its models as well as possible. We introduce several forgetting operators tailored to accomplish the removal of different pieces of the ABA knowledge base—assumptions, contraries, and atoms—formalizing a diverse selection of perspectives on this issue. We examine the quality of our operators by studying their compliance with suitable desiderata we propose. Thereby, we investigate the impact of the operators on the syntax of the given ABA knowledge base, its semantics, but also the instantiated argumentation framework; thus bridging recent forgetting studies on non-monotonic formalisms including argumentation theory.

1 Introduction

A highly relevant research direction in knowledge representation and reasoning is concerned with the investigation of dynamical environments, i.e. situations where we are given knowledge bases undergoing changes (Gabbay et al. 2021). Argumentation is an inherently dynamic process and it is therefore not surprising that in this field, researchers investigated the issue extensively. Several problems concerning dynamics have been considered like enforcing certain target arguments (Wallner, Niskanen, and Järvisalo 2016; Niskanen, Wallner, and Järvisalo 2018; Borg and Bex 2021), strong equivalence (Oikarinen and Woltran 2011), incorporating new beliefs (Falappa, Kern-Isberner, and Simari 2009; Haret, Wallner, and Woltran 2018) or repairing a knowledge base (Baumann and Ulbricht 2019).

Typical dynamic tasks are concerned with adding information to a knowledge base and how to deal with arising conflicts between old and new knowledge. Recently, however, there has also been an increasing amount of research on how arbitrary information can be removed, or forgotten, in such a way that the remaining information represented by the knowledge base is preserved. Forgetting was first conceived and resolved in the realm of classical logic (Lin and Reiter 1994). Later, this topic received broader attention in the area of knowledge representation and reasoning, most notably in of logic programming (LP). Due to their non-monotonicity, at first it was not clear which properties a "reasonable" forgetting operator should obey for LPs.

Researchers therefore developed several desiderata to investigate the behavior of forgetting operators (Eiter and Wang 2008; Zhang and Zhou 2009; Wang, Wang, and Zhang 2013; Wang et al. 2014; Knorr and Alferes 2014; Delgrande and Wang 2015). It became apparent that so called *strong persistence* captures best the essence of forgetting (Gonçalves, Knorr, and Leite 2016). Intuitively, *persistence* requires that the target atom that is supposed to be forgotten is removed from each model of the knowledge base; otherwise, the models remain unchanged. In addition, there has been a plethora of suggestions for concrete forgetting procedures satisfying different sets of desiderata (Zhang and Foo 2006; Eiter and Wang 2008; Knorr and Alferes 2014; Gonçalves et al. 2021; Berthold 2022), cf. (Berthold et al. 2019) for an overview.

More recently, forgetting was considered in abstract argumentation (Baumann, Gabbay, and Rodrigues 2020) and studied with respect to the forgetting properties in LPs as well as several new ones (Baumann and Berthold 2022). It turns out that forgetting from abstract argumentation frameworks (AFs) cannot be reduced to LPs, and that many of the desiderata proposed for LPs, including the commonly agreed persistence, are not feasible for AFs. Consequently forgetting in AFs has been analyzed with respect to alternative desiderata, including one which we coin *elimination* here. It requires all extensions with the forgotten arguments to be removed, thus eliminating their entire context.

The goal of the present paper is to bridge the two formalisms and to study forgetting in the context of assumption-based argumentation (ABA), which shares many common features with both LPs and AFs and is one of the primal structured argumentation formalisms (Toni 2014). As noticed in various contexts, pushing research from abstract to structured argumentation is a challenging endeavor (Wallner 2020; Rapberger and Ulbricht 2022; Prakken 2022). We will provide a thorough study of forgetting with respect to ABA, putting a particular focus on the two desiderata persistence and elimination.

An ABA knowledge base is composed of different features, namely so-called *assumptions*, their *contraries*, but also ordinary *atoms*. Therefore our study of forgetting in ABA covers several different aspects, all of which inducing their own intuition of what forgetting should mean within the respective context. Consider our running example.

Example 1.1. *A group of friends is having a discussion about vacation plans. They may tie their decision about whether to go on a city-trip or to the sea to the weather. In the city there are only expensive hotels because currently there is carnival season. At the sea there are also cheap options available. One of the friends, Antoine, does not have a lot of spare money and can therefore only join if a cheap hotel is chosen. There are three possible scenarios:*

- $\{good\_weather, sea, exp\_hotel\}$,
- $\{good\_weather, sea, cheap\_hotel, antoine\_joins\}$,
- $\{bad\_weather, city, exp\_hotel\}$.

Now let us assume that some participants do not like to swim. Therefore they do not consider going to the sea as part of the discussion. However, all the other aspects remain unchanged, i.e. going to the city is still tied to bad weather. In their eyes the possible outcomes are thus as follows.

- $\{good\_weather,$ ~~sea~~$, exp\_hotel\}$,
- $\{good\_weather,$ ~~sea~~$, cheap\_hotel, antoine\_joins\}$,
- $\{bad\_weather, city, exp\_hotel\}$.

We will see that updating a knowledge base in this way adheres to the so-called persistence *desideratum, requiring to remove a certain piece of information from each model.*

After a night to sleep the discussion over, a participant may decide that she indeed does not want to go on a trip at all, if there is bad weather. From her point of view, we need to update the knowledge base as follows.

- $\{good\_weather, sea, exp\_hotel\}$,
- $\{good\_weather, sea, cheap\_hotel, antoine\_joins\}$,
- ~~$\{bad\_weather, city, exp\_hotel\}$~~.

We will formalize this requirement with the elimination *desideratum that requires to delete certain models entirely.*

As a final example, let us suppose our group of friends does not want to go without Antoine. Unless the premises of our discussion change, e.g. by pooling money to pay for Antoine's stay as well, we arrive at the following models.

- ~~$\{good\_weather, sea, exp\_hotel\}$~~,
- $\{good\_weather, sea, cheap\_hotel, antoine\_joins\}$,
- ~~$\{bad\_weather, city, exp\_hotel\}$~~.

Formally, the participants remove the counter-argument for Antoine joining, i.e. the contrary *of $antoine\_joins$. We will cover such aspects in our discussion on contrary forgetting.*

In this work, we provide a solid theoretical foundation for limits and possibilities of forgetting different aspects of an ABA knowledge base. More specifically, the main contributions of this paper can be summarized as follows.

- We guide our investigation by proposing several desiderata which formalize the properties forgetting operators should have in different scenarios.
- We give (im)possibility results for satisfaction of our desiderata. Thereby we use both set-theoretical but also complexity-theoretic arguments, covering both the semantics of an ABA knowledge base as well as the instantiation procedure.
- Whenever possible, we directly construct intuitive forgetting operators satisfying the respective requirements.

2 Background

Abstract Argumentation. We fix a background set $\mathcal{U}$. An argumentation framework (AF) (Dung 1995) is a directed graph $F = (A, R)$, where $A \subseteq \mathcal{U}$ represents a set of arguments and $R \subseteq A \times A$ models *attacks* between them. By $A(F)$ and $R(F)$ we denote the arguments and attacks occurring in F, respectively. For a set $E \subseteq A$, we let $E^+_F = \{x \in A \mid \exists y \in E, (y, x) \in R\}$; also, E is *conflict-free* in F iff for no $x, y \in E$, $(x, y) \in R$. E *defends* an argument x iff E attacks each attacker of x. A conflict-free set E is *admissible* in F ($E \in ad(F)$) iff it defends all its elements. A *semantics* is a function $\sigma : \mathcal{F} \to 2^{2^{\mathcal{U}}}$ with $F \mapsto \sigma(F) \subseteq 2^A$; each $E \in \sigma(F)$ is called a σ *extension*. Here we consider so-called *complete*, *grounded*, *preferred*, and *stable* semantics (abbr. co, gr, pr, stb).

Definition 2.1. *Let $F = (A, R)$ be an AF and $E \in ad(F)$. Then $E \in co(F)$ iff E contains all arguments it defends; $E \in gr(F)$ iff E is $\subseteq$-minimal in $co(F)$; $E \in pr(F)$ iff E is $\subseteq$-maximal in $co(F)$; $E \in stb(F)$ iff $E^+_F = A \setminus E$.*

Assumption-based Argumentation. We assume a *deductive system* $(\mathcal{L}, \mathcal{R})$, where $\mathcal{L}$ is a set of atoms and $\mathcal{R}$ is a set of inference rules over $\mathcal{L}$. A rule $r \in \mathcal{R}$ has the form $a_0 \leftarrow a_1, \dots, a_n$, $a_i \in \mathcal{L}$; $head(r) = a_0$ is the head and $body(r) = \{a_1, \dots, a_n\}$ is the (possibly empty) body of r.

Definition 2.2. *An* ABA framework*, ABAF for short, is a tuple $\mathcal{D} = (\mathcal{L}, \mathcal{R}, \mathcal{A}, \overline{})$, where $(\mathcal{L}, \mathcal{R})$ is a deductive system, $\mathcal{A} \subseteq \mathcal{L}$ a set of assumptions, and $\overline{} : \mathcal{A} \to \mathcal{L}$ a contrary function.*

In this work, we focus on frameworks which are *flat*, i.e., $head(r) \notin \mathcal{A}$ for each rule $r \in \mathcal{R}$, and *finite*, i.e., $\mathcal{L}$, $\mathcal{R}$, $\mathcal{A}$ are finite; also, each rule is stated explicitly (given as input). By $\mathcal{L}(\mathcal{D})$, $\mathcal{R}(\mathcal{D})$ and $\mathcal{A}(\mathcal{D})$ we denote the atoms, rules and assumptions occurring in an ABAF $\mathcal{D}$, respectively. Similarly, by $\mathcal{A}(r)$ and $\mathcal{L}(r)$, we denote all assumptions, resp. atoms, appearing in a rule r.

An atom $p \in \mathcal{L}$ is *tree-derivable* from assumptions $S \subseteq \mathcal{A}$ and rules $R \subseteq \mathcal{R}$, denoted by $S \vdash_R p$, iff there is a finite rooted labeled tree t such that i) the root of t is labeled with p, ii) the set of labels for the leaves of t is equal to S or $S \cup \{\top\}$, and iii) for each internal node v of t there is a rule $r \in R$ such that v is labeled with $head(r)$ and labels of the children correspond to $body(r)$ or $\top$ iff $body(r) = \emptyset$.

By $Th_{\mathcal{D}}(S) = \{p \mid \exists S' \subseteq S : S' \vdash_R p\}$ we denote the set of all conclusions derivable from an assumption-set S in an ABAF $\mathcal{D}$. Observe that $S \subseteq Th_{\mathcal{D}}(S)$ since each $a \in \mathcal{A}$ is derivable from $\{a\} \vdash_\emptyset a$ (a tree with no internal node).

A set S of assumptions *attacks* a set T of assumptions iff $\overline{a} \in Th_D(S)$ for some $a \in T$; S is conflict-free iff it does not attack itself; S is admissible ($S \in ad(\mathcal{D})$) iff it defends itself, i.e. for any set T attacking S it holds that S counter-attacks T as well. We define grounded, complete, preferred, and stable ABA semantics (abbr. gr, co, pr, stb).

Definition 2.3. *Let $\mathcal{D} = (\mathcal{L}, \mathcal{R}, \mathcal{A}, \overline{})$ be an ABAF. Further, let $S \subseteq \mathcal{A}$ be admissible in $\mathcal{D}$. Then $S \in co(\mathcal{D})$ iff S contains every assumption it defends; $S \in gr(\mathcal{D})$ iff S is $\subseteq$-minimal in $co(\mathcal{D})$; $S \in pr(\mathcal{D})$ iff S is $\subseteq$-maximal in $co(\mathcal{D})$; $S \in stb(\mathcal{D})$ iff S attacks each $\{x\} \subseteq \mathcal{A} \setminus S$.*

We let $\Sigma = \{ad, co, gr, pr, stb\}$ be the set of all semantics considered in this paper. For a set S of assumptions, we let $\overline{S} = \{\overline{a} \mid a \in S\}$.

Definition 2.4. *The associated AF $F_{\mathcal{D}} = (A, R)$ of an ABAF $\mathcal{D} = (\mathcal{L}, \mathcal{R}, \mathcal{A}, \overline{})$ is given by $A = \{S \vdash p \mid \exists R \subseteq \mathcal{R} : S \vdash_R p\}$ and R, where $(S \vdash p, S' \vdash p') \in R$ iff $p \in \overline{S'}$.*

ABA semantics can be interpreted by means of extensions in the corresponding AF $F_{\mathcal{D}}$ as shown in (Čyras et al. 2018).

Proposition 2.5. *Given an ABAF $\mathcal{D} = (\mathcal{L}, \mathcal{R}, \mathcal{A}, \overline{})$, its corresponding AF F and a semantics $\sigma \in \Sigma$. If $E \in \sigma(F)$ then $\bigcup_{S \vdash p \in E} S \in \sigma(\mathcal{D})$; if $S \in \sigma(\mathcal{D})$ then $\{S' \vdash p \mid \exists S' \subseteq S, R \subseteq \mathcal{R} : S' \vdash_R p\} \in \sigma(F)$.*

Example 2.6. *We formalize our introductory example: let $\mathcal{D} = (\mathcal{L}, \mathcal{R}, \mathcal{A}, \overline{})$ with $\mathcal{A} = \{exp\_hotel, cheap\_hotel, antoine\_joins, sea, city, good\_weather, bad\_weather\}$, $\mathcal{L} = \mathcal{A} \cup \{exp\_hotel_c, cheap\_hotel_c, antoine\_joins_c, sea_c, city_c, good\_weather_c, bad\_weather_c, carnival\}$ and contraries $\overline{x} = x_c$ for each $x \in \mathcal{A}$. The relationships between these atoms, such as that good and bad weather are mutually exclusive, are formalized by the following rules $\mathcal{R}$:*

$$
\begin{array}{ll}
bad\_weather_c \leftarrow good\_weather & sea_c \leftarrow bad\_weather \\
good\_weather_c \leftarrow bad\_weather & city_c \leftarrow good\_weather \\
exp\_hotel_c \leftarrow cheap\_hotel & cheap\_hotel_c \leftarrow exp\_hotel \\
cheap\_hotel_c \leftarrow city, carnival & carnival \leftarrow \\
antoine\_joins_c \leftarrow exp\_hotel &
\end{array}
$$

We obtain the following attacks between assumptions (we omit attacks between any super-sets since the singletons already characterize the attack structure of the framework, that is, assumption-set S attacks assumption-set T iff there are $a \in S$, $b \in T$, such that $\{a\}$ attacks $\{b\}$).

$$\begin{tikzcd}
 & \{good\_weather\} \arrow[rr, leftrightarrow] \arrow[dl, bend right] & & \{bad\_weather\} \arrow[dr, bend left] & \\
\{city\} \arrow[dr, bend right] & & \{antoine\_joins\} & & \{sea\} \\
 & \{cheap\_hotel\} \arrow[rr, leftrightarrow] & & \{exp\_hotel\} \arrow[ul, bend right] &
\end{tikzcd}$$

The preferred extensions of our ABAF $\mathcal{D}$ are given as the assumption-sets $\{good\_weather, exp\_hotel, sea\}$, $\{good\_weather, cheap\_hotel, antoine\_joins, sea\}$, as well as $\{bad\_weather, city, exp\_hotel\}$.

3 Forgetting in ABA

Let us start by stipulating what we mean by a forgetting operator. Within the scope of this work, we focus on forgetting i) an ordinary atom, ii) an assumption, or iii) a contrary. All of them occur in the language $\mathcal{L}$ of a given ABAF. Thus a forgetting operator takes an ABAF $\mathcal{D}$ together with an element of $p \in \mathcal{L}$ as input, and returns an ABAF $\mathsf{f}(\mathcal{D}, p)$ with the intuitive meaning that p is forgotten.

Definition 3.1. *Let $\mathcal{ABA}$ be the set of all ABAFs, and $\mathcal{ATOM}$ the set of all atoms. A* forgetting operator f *is a (partial) mapping $\mathsf{f} : \mathcal{ABA} \times \mathcal{ATOM} \to \mathcal{ABA}$.*

That is, we do not impose any restriction on the behavior of f. The following example shall illustrate the variety of conceivable forgetting operators.

Example 3.2. *i) The most trivial forgetting operator is the identity, i.e. $\mathsf{f}_{id}(\mathcal{D}, p) = \mathcal{D}$ for each ABAF $\mathcal{D}$ and p an atom.*

ii) Another basic approach to "forget" an atom is by simply removing it from the knowledge base and all rules it occurs in. Given $\mathcal{D} = (\mathcal{L}, \mathcal{R}, \mathcal{A}, \overline{})$ and $p \in \mathcal{L}$ we let

$$\mathsf{f}_{basic}(\mathcal{D}, p) := \left(\mathcal{L} \setminus \{p\}, \mathcal{R}', \mathcal{A} \setminus \{p\}, \overline{}|_{\mathcal{A} \setminus \{p\}}\right)$$

with rules

$$\mathcal{R}' = \{head(r) \leftarrow body(r) \setminus \{p\} \mid r \in \mathcal{R},\ head(r) \neq p\}.$$

iii) A somewhat more interesting approach is the following: Assuming $p \in \mathcal{A}$, let us forget that p is an assumption, i.e. we do not remove it from $\mathcal{L}$, but we change $\mathcal{D}$ in a way that p is not defeasible anymore. To this end we set

$$\mathsf{f}_{strict}(\mathcal{D}, p) := \left(\mathcal{L}, \mathcal{R} \cup \{p \leftarrow\}, \mathcal{A} \setminus \{p\}, \overline{}|_{\mathcal{A} \setminus \{p\}}\right).$$

Recall that within the scope of this study we consider forgetting an assumption, a contrary or an ordinary atom. However, we want to avoid situations where an assumption a is also a contrary and thus forgetting a would correspond to both situations simultaneously. We therefore stipulate within this paper that $\mathcal{A} \cap \overline{\mathcal{A}} = \emptyset$. We do not lose any expressive power by this restriction, as the following result formalizes.

Proposition 3.3. *For any ABA $\mathcal{D}$ and $\sigma \in \Sigma$, there is an ABA $\mathcal{D}' = (\mathcal{L}, \mathcal{R}, \mathcal{A}, \overline{})$ s.t. $\sigma(\mathcal{D}) = \sigma(\mathcal{D}')$ and $\mathcal{A} \cap \overline{\mathcal{A}} = \emptyset$.*

As we already mentioned, a forgetting operator is in general an arbitrary function. The naturally arising question is therefore: Are the aforementioned operators useful? Are they well-behaved? In which scenarios is their output satisfactory? In order to formally address questions of this kind, research on forgetting is driven by various desiderata. Their goal is to i) investigate limits and possibilities of forgetting operators in general and ii) formalize the behavior of given forgetting operators in the corresponding situation. Consequently, the starting point of study will be the development of ABA forgetting desiderata. In the following, we discuss on which aspects our desiderata will focus. We want to emphasize that our desiderata are intentionally strong, oftentimes not satisfiable simultaneously and sometimes even not on their own. Consequently, we do not claim that a certain forgetting operator is "bad", "counter-intuitive", or "unsuitable" when not satisfying certain desiderata. Rather, the properties we consider are tailored to examine the edge of satisfiability, thereby providing an investigation of what can possibly be expected from an operator and what not.

Syntax. A rather basic requirement is that once the information is forgotten, it does indeed not occur in the knowledge base anymore, in a mere syntactical sense.

Desideratum 3.4. *Given an ABAF $\mathcal{D}$ and an atom $p \in \mathcal{L}$. A forgetting operator f satisfies*

$(\mathbf{D})$ deletion *iff* $\mathcal{L}(\mathsf{f}(\mathcal{D}, p)) \subseteq \mathcal{L}(\mathcal{D}) \setminus \{p\}$.

This desideratum is of course rather straightforward to satisfy on its own. For example, it is easy to see that f_{basic} from Example 3.2 satisfies $(\mathbf{D})$. It is not satisfied by f_{strict} though since here the atom is only removed from $\mathcal{A}$, not $\mathcal{L}$.

Semantics. Such syntactical removal of items to be forgotten is often considered as a baseline desideratum in the literature, e.g. in the context of forgetting in logic programs. However, forgetting was initially only perceived as a semantic notion. The paper (Lin and Reiter 1994) even introduced forgetting operators that add explicit mentions of facts to be forgotten to first order formulas. In this work, we follow this approach and mainly focus on the *semantical* implications of forgetting a target information.

A forgetting notion that is rather dominant in the literature is that of *persistence* which requires a forgotten atom to be cut out of all extensions. We will investigate this desideratum and show that in general, it is unfortunately too strong to be applied to ABA. An alternative notion of forgetting is that of *elimination*. It requires all extensions mentioning the forgotten assumption to be removed. However, the exact formulation of the semantical desiderata depends on the situation under consideration – ordinary atoms, assumptions, or contraries – and we will phrase them suitably within the corresponding sections.

Arguments. A special feature of structured argumentation formalisms like ABA is that apart from the given knowledge base, there is also an induced argumentation framework under consideration. We therefore also want to investigate the impact of forgetting on the constructed arguments and their interactions. As in the case of the semantical desiderata, this depends on the situation at hand.

Computation. Even the most powerful desiderata are pointless if we cannot compute the forgetting operator f with reasonable effort. For example, what if f satisfies many desirable properties, but computation of f requires explicit knowledge of the semantics of $\mathcal{D}$? Then, computing $\mathsf{f}(\mathcal{D}, p)$ would require an intractable task as a pre-processing step. Or what if $\mathsf{f}(\mathcal{D}, p)$ is exponential in the size of $\mathcal{D}$? Then we bloat $\mathcal{D}$ although our intention was to "forget" something. There are many conceivable ways to prevent such counter-intuitive scenarios. Within the scope of this paper however we restrict our attention to the most basic requirement, stipulating that $\mathsf{f}(\mathcal{D}, p)$ can be computed in polynomial time.

Definition 3.5. *Given an ABAF $\mathcal{D}$ and an atom p, a forgetting operator f is* tractable *iff there is a polynomial-time algorithm which computes $\mathsf{f}(\mathcal{D}, p)$.*

Within the present study we will refrain from constructing non-tractable forgetting operators. The reason is that doing so would call for an empirical evaluation witnessing feasibility of the approach. This is however beyond the scope of our theoretical analysis.

4 Warm-Up: Forgetting an Atom

In this section we will start with the simple case of forgetting an ordinary atom $p \in \mathcal{L} \setminus (\mathcal{A} \cup \overline{\mathcal{A}})$. Such atoms facilitate derivations, but do not interact with the extensions directly. Hence we expect rather strong results in the context of this section: It should be possible to get rid of p as above without disrupting the remainder of the knowledge base.

For ease of presentation, within this section we assume that our operator f is defined for $p \in \mathcal{L} \setminus (\mathcal{A} \cup \overline{\mathcal{A}})$ only.

4.1 Atom Forgetting Desiderata

A notable feature of the way ABA semantics are defined is that the ordinary atoms $p \in \mathcal{L} \setminus (\mathcal{A} \cup \overline{\mathcal{A}})$ are not considered in the extensions which serve as the models of an ABA knowledge base. More specifically, if $E \in \sigma(D)$, then $p \notin E$ since $E \subseteq \mathcal{A}$. Consequently, in the case of atom forgetting it is even conceivable to remove p from the given ABAF without changing the semantics at all. This leads to the following, quite unique desideratum.

Desideratum 4.1. *Given an ABAF $\mathcal{D}$, an atom $p \in \mathcal{L} \setminus (\mathcal{A} \cup \overline{\mathcal{A}})$, a semantics σ. A forgetting operator f satisfies*

$(\mathbf{S})$ steadiness *iff* $\sigma(\mathsf{f}(\mathcal{D}, p)) = \sigma(\mathcal{D})$.

Example 4.2. *Although the carnival in our introductory example has an effect on the prices of the hotels in the city and therefore also an effect on the outcome of the discussion, we may want to dismiss this information, since it is only implicit, and not directly part of the outcome of the discussion.*

We observe that steadiness on its own is not a meaningful desideratum since the identity $\mathsf{f} = id$ already satisfies $(\mathbf{S})$. From an intuitive point of view it is therefore apparent that we require meaningful forgetting operators to satisfy at least one further desideratum, e.g. deletion $(\mathbf{D})$ which would already suffice to rule out trivial forgetting operators.

As we already mentioned, we also want to consider the AF induced by $\mathcal{D}$. We expect that forgetting p results in a situation where no agent can argue for p anymore. However, as the remaining parts of the knowledge base shall be unaffected, we do not seek to alter the other arguments. Recall that $A(F)$ denote the arguments occurring in an AF F.

Desideratum 4.3. *Given an ABAF $\mathcal{D}$ and atom $p \in \mathcal{L} \setminus (\mathcal{A} \cup \overline{\mathcal{A}})$. A forgetting operator f satisfies*

$(\mathbf{AD})$ argument deletion *iff*

$$A(F_{\mathsf{f}(\mathcal{D},p)}) = \{A \vdash s \in A(F_{\mathcal{D}}) \mid s \neq p\}.$$

We want to emphasize that the attacks in $F_{\mathcal{D}}$ are determined by the assumptions and conclusions of the tree-based arguments; thus argument deletion $(\mathbf{AD})$ does not alter the attack structure within the AF, only the occuring arguments.

4.2 The Replace Operator

In this subsection we will construct our first meaningful forgetting operator. In a nutshell, the idea is to bridge the target atom p in each rule where it occurs. This way, we ensure that the semantics of $\mathcal{D}$ do not alter, but p is not required anymore for the derivations. The forgetting operator we obtain this way will satisfy deletion $(\mathbf{D})$, argument deletion $(\mathbf{AD})$, and steadiness $(\mathbf{S})$ simultaneously. In addition, it is possible to compute it in polynomial time.

Formally, consider the following operator rpl (replace): Given some atom p we let

$$\begin{aligned} rpl(\mathcal{R}, p) = &\{head(r) \leftarrow (body(r) \cup body(r')) \setminus \{p\} \\ &\mid r, r' \in \mathcal{R}, p = head(r') \in body(r)\}, \end{aligned}$$

i.e. any appearance of p within a rule body is replaced with the body of a rule with p in its head.

We define our forgetting operator by replacing the rules with appearances of p with these new rules.

Definition 4.4. *Let $\mathcal{D} = (\mathcal{L}, \mathcal{R}, \mathcal{A}, ^{-})$ be an ABAF and $p \in \mathcal{L} \setminus (\mathcal{A} \cup \overline{\mathcal{A}})$. Let $\mathsf{f}_l(\mathcal{D}, p) := (\mathcal{L} \setminus \{p\}, \mathcal{R}', \mathcal{A}, ^{-})$, where*

$$\mathcal{R}' := \mathcal{R} \setminus \{r \in \mathcal{R} \mid p \in \mathcal{L}(r)\} \cup rpl(\mathcal{R}, p).$$

As desired, f_l bridges each occurrence of p of any tree derivation in order to ensure that each argument tree can still be derived analogously, but without explicit entailment of p.

Example 4.5. *In our running example the atom carnival appears in the head of one rule (with empty rule body). If we apply rpl to the set $\mathcal{R}$ and 'carnival', in this special case, 'carnival' is replaced by nothing, i.e. removed from all rule bodies. We have $\mathcal{R}'$:*

$$
\begin{array}{ll}
bad\_weather_c \leftarrow good\_weather & sea_c \leftarrow bad\_weather \\
good\_weather_c \leftarrow bad\_weather & city_c \leftarrow good\_weather \\
exp\_hotel_c \leftarrow cheap\_hotel & cheap\_hotel_c \leftarrow exp\_hotel \\
cheap\_hotel_c \leftarrow city, \cancel{carnival} & \cancel{carnival \leftarrow} \\
antoine\_joins_c \leftarrow exp\_hotel &
\end{array}
$$

By definition, p does not occur in $\mathsf{f}_l(\mathcal{D}, p)$. This also implies that no argument in $\mathsf{f}_l(\mathcal{D}, p)$ entails it. Therefore, deletion $(\mathbf{D})$ and argument deletion $(\mathbf{AD})$ are satisfied.

Proposition 4.6. *Given an ABAF $\mathcal{D}$ and an atom $p \in \mathcal{L} \setminus (\mathcal{A} \cup \overline{\mathcal{A}})$. Then $\mathsf{f}_l(\mathcal{D}, p)$ satisfies both $(\mathbf{D})$ and $(\mathbf{AD})$.*

Let us now turn our attention to the semantics. As the following lemma formalizes, the assumptions sets in $\mathcal{D}$ and $\mathsf{f}_l(\mathcal{D}, p)$ agree on what they can entail, except p.

Lemma 4.7. *Let $\mathcal{D} = (\mathcal{L}, \mathcal{R}, \mathcal{A}, ^{-})$ be an ABAF and $p \in \mathcal{L} \setminus \mathcal{A}$. Let $S \subseteq \mathcal{A}$. Then $Th_{\mathsf{f}_l(\mathcal{D},p)}(S) = Th_{\mathcal{D}}(S) \setminus \{p\}$.*

This implies that f_l satisfies steadiness $(\mathbf{S})$ as well. We also observe that f_l can be computed in polynomial time, implying tractability. To summarize, f_l is capable of satisfying all atom forgetting desiderata.

Theorem 4.8. *If $p \in \mathcal{L} \setminus (\mathcal{A} \cup \overline{\mathcal{A}})$, then the forgetting operator f_l satisfies $(\mathbf{D})$, $(\mathbf{AD})$, $(\mathbf{S})$ and is tractable.*

5 Forgetting an Assumption

In the previous section, our forgetting operator f_l was quite successful. This is not too surprising since ordinary atoms only have a moderate impact on the structure of (the models of) $\mathcal{D}$. In this section we turn to our main subject of investigation: forgetting an assumption. Assumptions are the central protagonists in an ABAF and removing them while trying to ensure an intuitive outcome turns out to be a challenging endeavor.

The assumption forgetting desiderata we will develop are not simultaneously satisfiable (see Theorem 5.24). Indeed, in many cases, they are not even satisfiable on their own. Due to this observation, after formalizing our desiderat we will devote a subsection to each of them. Each time, we start with a theoretical analysis in order to examine if the desideratum under consideration is satisfiable – and if so, if there is a tractable operator which does the job. In case we can answer these questions affirmatively, we will proceed by constructing suitable operators.

For ease of presentation, within this section we implicitly assume that our forgetting operator f is defined for assumptions $a \in \mathcal{A}$ only.

5.1 Assumption Forgetting Desiderata

One of the primal desiderata in forgetting in the literature is *persistence*. It formalizes that the target conclusion shall be removed from each extension, but otherwise the models of the given knowledge base persist. In the context of our motivating Example 1.1, this corresponds to the situation where some participants forget about the sea since they are not interested in swimming – the possible outcomes of the discussion remain the same, except no extension contains "sea" anymore.

Desideratum 5.1. *Given an ABAF $\mathcal{D}$, assumption a, and semantics σ. A forgetting operator f satisfies*

$(\mathbf{P})$ persistence *iff* $\sigma(\mathsf{f}(\mathcal{D}, a)) = \{E \setminus \{a\} \mid E \in \sigma(\mathcal{D})\}$.

Example 5.2. *Applied to the assumption sea, the persistence $(\mathbf{P})$ desideratum formalizes the view point of those participants who do not consider going to the beach as part of the discussion at all. Adjusting the knowledge base according to their plans would require an ABAF with preferred extensions $\{good\_weather, exp\_hotel\}$, $\{good\_weather, cheap\_hotel, antoine\_joins\}$, as well as $\{bad\_weather, city, exp\_hotel\}$.*

Another conceivable way to forget is to remove extensions containing a entirely, leading to so-called *elimination*. This corresponds to the situation in Example 1.1 where some participant decides that she does not want to join if there is bad weather. We therefore eliminate each extension containing some target atom. If the atom is contained in each extension, we require the resulting ABAF to be trivial. In order not to lose satisfiability in particular corner cases, we relax elimination when a skeptically accepted assumption is forgotten under universally defined semantics. We define elimination in ABA as follows.

Desideratum 5.3. *Given an ABAF $\mathcal{D}$, assumption a, and semantics σ. A forgetting operator f satisfies*

$(\mathbf{E})$ elimination *iff*

$$\sigma(\mathsf{f}(\mathcal{D}, a)) = \begin{cases} \{E \in \sigma(\mathcal{D}) \mid a \notin E\} & \textit{if } a \notin \bigcap \sigma(\mathcal{D}), \\ \{\emptyset\} & \textit{if } \sigma \neq stb, a \in \bigcap \sigma(\mathcal{D}), \\ \emptyset & \textit{if } \sigma = stb, a \in \bigcap \sigma(\mathcal{D}). \end{cases}$$

Example 5.4. *In our Example 2.6, Elimination formalizes the situation after the group decides to only go on vacation when there is no bad weather. Adjusting the knowledge base according to their plans would require an ABAF with preferred extensions $\{good\_weather, exp\_hotel, sea\}$ and $\{good\_weather, cheap\_hotel, antoine\_joins, sea\}$.*

Let us now turn our attention towards the effect on the argumentation structure caused by our forgetting procedure. We may formulate similarly motivated desiderata about the structured arguments of the induced framework.

Argument persistence $(\mathbf{AP})$ formalizes that an agent does not rely on the forgotten assumption a anymore, in order to bring forward an argument. That is, the arguments that can be constructed are analogous, but a is not mentioned in the debate corresponding to $\mathsf{f}(\mathcal{D}, a)$. This follows the same rationale as persistence $(\mathbf{P})$, but from the point of view of the exchange of arguments.

Desideratum 5.5. *Given an ABAF $\mathcal{D}$ and assumption a. A forgetting operator* f *satisfies*

$(\mathbf{AP})$ argument persistence *iff*

$$A(F_{\mathsf{f}(\mathcal{D},a)}) = \{A \setminus \{a\} \vdash s \mid A \vdash s \in A(F_{\mathcal{D}})\}.$$

Analogously, we consider argument elimination $(\mathbf{AE})$ as a counterpart to elimination $(\mathbf{E})$. Here, we exclude arguments relying on a. That is, in the debate corresponding to $\mathsf{f}(\mathcal{D},a)$ it is impossible to bring forward these arguments.

Desideratum 5.6. *Given an ABAF $\mathcal{D}$ and assumption a. A forgetting operator* f *satisfies*

$(\mathbf{AE})$ argument elimination *iff*

$$A(F_{\mathsf{f}(\mathcal{D},a)}) = \{A \vdash s \in A(F_{\mathcal{D}}) \mid a \notin A\}.$$

Note that in our definition of $(\mathbf{AE})$ we do not have to require $s \neq a$ explicitly when forgetting a since our ABAFs are flat, i.e. $a \notin A$ in $A \vdash s$ implies $s \neq a$.

5.2 Persistent Forgetting

This subsection is devoted to satisfaction of persistence $(\mathbf{P})$.

Theoretical Analysis. It turns out that, when forgetting an assumption, persistence alone (without any other requirement) is already unsatisfiable for all but gr semantics. The intuitive reason for pr and stb semantics is that $pr(\mathcal{D})$ and $stb(\mathcal{D})$ form antichains and this property might get lost when removing the target assumption from each extension.

Example 5.7. *Assume we are given any semantics $\sigma \in \{stb, pr\}$, and an ABAF $\mathcal{D} = (\mathcal{L}, \mathcal{R}, \mathcal{A}, ^-)$, where*

$$\mathcal{L} = \{a, b, a_c, b_c\} \qquad \mathcal{R} = \{a_c \leftarrow b;\;\; b_c \leftarrow a\}$$
$$\mathcal{A} = \{a, b\} \qquad\qquad ^- = \{(a, a_c), (b, b_c)\}$$

Then $\sigma(\mathcal{D}) = \{\{a\}, \{b\}\}$. In order for a forgetting operator f *to satisfy persistence when forgetting b, we would have to have $\sigma(\mathsf{f}(\mathcal{D}, a)) = \{\{a\}, \emptyset\}$. There is however no ABAF with these stable resp. preferred extensions:*

- *If $\{a\}$ is preferred, i.e. maximally admissible, then $\emptyset$ is certainly not maximal. Thus, $pr(\mathsf{f}(\mathcal{D}, b)) = \{\{a\}, \emptyset\}$ is impossible.*
- *If $\emptyset$ is stable, i.e. attacks each assumption, then $\{a\}$ is certainly not conflict-free. Thus, $stb(\mathsf{f}(\mathcal{D}, b)) = \{\{a\}, \emptyset\}$ is impossible.*

In this previous example, we focused on pr and stb semantics. However, we can make similar observations for ad and co semantics as well. Put simply, while pr and stb semantics form antichains, there are also characterizing properties for the ad and co extensions of an ABAF $\mathcal{D}$ – and in general they get lost when trying to forget an assumption s.t. persistence $(\mathbf{P})$ holds. However, a comprehensive discussion of all technical details is beyond the scope of this work. We refer the reader to the technical supplement where the relevant proofs can be found.

On the other hand, we do not face such an issue for gr semantics. As is known, there is always a unique grounded extension. Now, given $G \in gr(\mathcal{D})$ it is certainly possible to construct $\mathsf{f}(\mathcal{D}, a)$ in a way that $G \setminus \{a\}$ becomes the new grounded extension (we refer the skeptical reader to Definition 5.11). We therefore end up with the following (im)possibility result for satisfiability of persistence $(\mathbf{P})$.

Theorem 5.8. *Let $\mathcal{D} = (\mathcal{L}, \mathcal{R}, \mathcal{A}, ^-)$ be an ABAF, $\sigma \in \Sigma$, and $a \in \mathcal{A}$. There is a forgetting operator* f *satisfying* $(\mathbf{P})$ *iff $\sigma = gr$.*

Since this subsection is driven by satisfaction of persistence, we will continue our investigation with grounded semantics and refrain from the remaining cases until our discussion on elimination $(\mathbf{E})$ forgetting.

Construction of Forgetting Operators. According to our theoretical results, we should be able to come up with a tractable forgetting operator for persistent forgetting under gr semantics. The idea is to remove the assumption a we want to forget from the body of each rule in $\mathcal{D}$, but only if a occurs in the grounded extension of $\mathcal{D}$. Towards investigating our operator, let us note some general observation which we find interesting on its own, independent of the context of forgetting. For this, consider the following notion.

Definition 5.9. *Given $\mathcal{D} = (\mathcal{L}, \mathcal{R}, \mathcal{A}, ^-)$ and $a \in \mathcal{A}$ we let $rm(\mathcal{D}, a) := (\mathcal{L}', \mathcal{R}', \mathcal{A}', ^-|_{\mathcal{A}'})$ where $\mathcal{A}' = \mathcal{A} \setminus \{a\}$, $\mathcal{L}' = \mathcal{L} \setminus \{a\}$, and $\mathcal{R}' = \{head(r) \leftarrow body(r) \setminus \{a\} \mid r \in \mathcal{R}\}$.*

Any σ extension $E \in \sigma(\mathcal{D})$ containing the assumption a survives the transition to the ABAF $rm(\mathcal{D}, a)$, where a is removed from each rule body. The intuitive reason is that in $rm(\mathcal{D}, a)$, the assumption a is not required anymore to entail the respective conclusions.

Lemma 5.10. *Let $\mathcal{D} = (\mathcal{L}, \mathcal{R}, \mathcal{A}, ^-)$ be an ABAF, $\sigma \in \{ad, co, pr, gr, stb\}$, and $a \in \mathcal{A}$. Moreover, suppose $a \in E$ for $E \in \sigma(\mathcal{D})$. Then $E \setminus \{a\} \in \sigma(rm(\mathcal{D}, a))$.*

This observation can be exploited in order to construct a forgetting operator for gr semantics: If a does not occur in the grounded extension, persistence $(\mathbf{P})$ does not require us to modify $\mathcal{D}$. Otherwise, we simply apply rm from above.

Definition 5.11. *Given $\mathcal{D} = (\mathcal{L}, \mathcal{R}, \mathcal{A}, ^-)$ and $a \in \mathcal{A}$ we let*

$$\mathsf{f}_{(\mathbf{P})gr}(\mathcal{D}, a) := \begin{cases} \mathcal{D} & \text{if } a \notin G \in gr(\mathcal{D}), \\ rm(\mathcal{D}, a) & \text{if } a \in G \in gr(\mathcal{D}). \end{cases}$$

Example 5.12. *Let $\mathcal{D} = (\mathcal{L}, \mathcal{R}, \mathcal{A}, ^-)$, where*

$$\mathcal{L} = \{a, b, c, a_c, b_c, c_c\} \quad \mathcal{R} = \{c_c \leftarrow b;\;\; b_c \leftarrow a\}$$
$$\mathcal{A} = \{a, b, c\} \qquad\qquad ^- = \{(a, a_c), (b, b_c), (c, c_c)\}$$

To forget a from $\mathcal{D}$ via $\mathsf{f}_{(\mathbf{P})gr}$, we first check whether a is in the unique ground extension – $gr(\mathcal{D}) = \{\{a, c\}\}$ – it is. We are hence in the second case and apply $rm(\mathcal{D}, a) = (\mathcal{L}', \mathcal{R}', \mathcal{A}', ^{-\prime})$, where

$$\mathcal{L}' = \{b, c, a_c, b_c, c_c\} \qquad \mathcal{R}' = \{c_c \leftarrow b;\;\; b_c \leftarrow\}$$
$$\mathcal{A}' = \{b, c\} \qquad\qquad ^{-\prime} = \{(b, b_c), (c, c_c)\}$$

We summarize the properties of $\mathsf{f}_{(\mathbf{P})gr}$.

Theorem 5.13. *The operator $\mathsf{f}_{(\mathbf{P})gr}$ is tractable. Moreover, it satisfies $(\mathbf{P})$ for gr semantics.*

However, the construction in Definition 5.11 does not satisfy deletion $(\mathbf{D})$. Indeed, it is not guaranteed that the assumption which we aim to forget is no longer part of the modified knowledge base. This issue can be circumvented via the following simple construction:

Definition 5.14. *Given $\mathcal{D} = (\mathcal{L}, \mathcal{R}, \mathcal{A}, ^{-})$ with $G \in gr(\mathcal{D})$ and $a \in \mathcal{A}$ we let $G' = G \setminus \{a\}$ and define*

$$\mathsf{f}_{(\mathbf{D})gr}(\mathcal{D}, a) := (G' \cup \overline{G}', \emptyset, G', ^{-}|_{G'}),$$

where $\overline{G}' = \{\overline{g} \mid g \in G'\}$ contains all contraries of G'.

The operator computes the grounded extension G (in polynomial time) and constructs a framework which contains only $G \setminus \{a\}$ without any rules. In contrast to $\mathsf{f}_{(\mathbf{P})gr}$, the operator $\mathsf{f}_{(\mathbf{D})gr}$ satisfies deletion.

Theorem 5.15. *The operator $\mathsf{f}_{(\mathbf{D})gr}$ satisfies $(\mathbf{D})$ and is tractable. Moreover, it satisfies $(\mathbf{P})$ for gr semantics.*

5.3 Elimination Forgetting

This subsection is devoted to satisfaction of elimination $(\mathbf{E})$.

Theoretical Analysis. As before, we start the formal investigation with the underlying satisfiability results. As it will turn out, elimination is a bit easier to satisfy than persistence. First of all, it is possible to satisfy elimination $(\mathbf{E})$ for all semantics except co.

Theorem 5.16. *Let $\mathcal{D} = (\mathcal{L}, \mathcal{R}, \mathcal{A}, ^{-})$ be an ABAF, $\sigma \in \Sigma$, and $a \in \mathcal{A}$. There is a forgetting operator f satisfying $(\mathbf{E})$ iff $\sigma \neq co$.*

Comparing Theorem 5.16 and Theorem 5.8 the question arises why persistence is so much harder to satisfy? The intuitive reason is that persistence $(\mathbf{P})$ requires us to modify extensions, i.e. delve into and change the structure of the given models. Elimination $(\mathbf{E})$ on the other hand can be satisfied by simply excluding certain extensions, i.e. posing additional constraints on the given models. This observation is also utilized by our operator from Definition 5.20 which relies on constraining $\sigma(\mathcal{D})$ in a suitable way.

Continuing our theoretical analysis, let us turn to tractability, i.e. for which semantics can elimination forgetting be realized by means of a polynomial-time computable operator? To this end we make the following observation regarding pr semantics. Suppose we are given an ABAF $\mathcal{D}$ with $ad(\mathcal{D}) = \{\emptyset, \{a, b\}, \{a, b, c\}\}$, i.e. $\{a, b, c\}$ is the unique preferred extension. Elimination forgetting a would require us to remove the whole extension, i.e. $pr(\mathcal{D}) = \{\emptyset\}$. But how can we ensure this? Simply removing $\{a, b, c\}$ would then imply that $\{a, b\}$ is now maximal, yielding $pr(\mathsf{f}(\mathcal{D}, a)) = \{\{a, b\}\}$ which is not the desired outcome.

While it is possible to circumvent this issue, it would require us to be aware of *each* admissible set in $\mathcal{D}$. However, the set of extensions is no part of the input of our forgetting operator and thus it is hardly conceivable that this is possible in polynomial time. We can formalize this observation in the following theorem, stating that elimination forgetting $(\mathbf{E})$ is only possible for pr semantics when making use of an intractable operator f.

Theorem 5.17. *Unless $\mathsf{coNP} = \Pi_{\mathsf{P}}^{2}$, there is no forgetting operator satisfying $(\mathbf{E})$ that is tractable for $\sigma = pr$.*

Indeed, the proof of this result relies on the fact that skeptical reasoning with preferred semantics is hard due to the maximization incorporated in the definition (Dvorák and Dunne 2018).

Construction of Forgetting Operators. In contrast to the previous subsection, it is this time possible to find a polynomial-time computable forgetting operator for stb and ad semantics. As we already mentioned, the underlying idea is to constrain the given extensions $\sigma(\mathcal{D})$. To this end we turn the target assumption a which shall be forgotten into a self-attacker. This renders any extension $E \in \sigma(\mathcal{D})$ with $a \in E$ self-conflicting. The remaining admissible resp. stable extension remain unaffected by this modification.

Before doing so, however, we have to take care of one additional hurdle: It might be the case that a shares a contrary with some other assumption b, i.e. $\overline{a} = \overline{b}$ which would undermine our idea, because we do not want to introduce additional attacks to b when forgetting a. To deal with this, consider the following pre-processing step which ensures that the contrary of a is not shared.

Definition 5.18. *Given an ABAF $\mathcal{D} = (\mathcal{L}, \mathcal{R}, \mathcal{A}, ^{-})$ and assumption a we let a_c be a fresh atom. Define $uc(\mathcal{D}, a) := (\mathcal{L}', \mathcal{R}', \mathcal{A}, ^{-'})$, where $\mathcal{L}' := \mathcal{L} \cup \{a_c\}$ and*

$$\mathcal{R}' := \mathcal{R} \cup \{a_c \leftarrow \overline{a}\} \qquad ^{-'} := {}^{-}|_{\mathcal{A}\setminus\{a\}} \cup (a, a_c)$$

Our operator $uc(\mathcal{D}, a)$ (unique contrary) does not alter the semantics of $\mathcal{D}$ and simply ensures that no assumption apart from a has the same contrary.

Lemma 5.19. *For any ABAF, assumption a, and semantics $\sigma \in \Sigma$, we have $\sigma(\mathcal{D}) = \sigma(uc(\mathcal{D}, a))$.*

Having ensured that a does not share the contrary, we can now provide our forgetting operator for elimination $(\mathbf{E})$ w.r.t. ad and stb semantics.

Definition 5.20. *Given $\mathcal{D} = (\mathcal{L}, \mathcal{R}, \mathcal{A}, ^{-})$ and $a \in \mathcal{A}$ we let $self(\mathcal{D}, a) := (\mathcal{L}, \mathcal{R} \cup \{\overline{a} \leftarrow a\}, \mathcal{A}, ^{-})$ and define*

$$\mathsf{f}_{(\mathbf{E})adstb}(\mathcal{D}, a) = self(uc(\mathcal{D}, a), a).$$

Thus, $\mathsf{f}_{(\mathbf{E})adstb}$ ensures that the contrary of a can be entailed once a is in our extension-set – hence enforcing a conflict. Extensions not containing a remain unaffected.

Example 5.21. *We consider a slightly modified version of Example 5.7. Let $\mathcal{D} = (\mathcal{L}, \mathcal{R}, \mathcal{A}, ^{-})$, where*

$$\mathcal{L} = \{a, b, c, a_c, b_c, c_c\} \quad \mathcal{R} = \{a_c \leftarrow b, c;\ \ b_c \leftarrow a\}$$
$$\mathcal{A} = \{a, b, c\} \quad ^{-} = \{(a, a_c), (b, b_c), (c, c_c)\}$$

We have $ad(\mathcal{D}) = \{\emptyset, \{a\}, \{b, c\}\}$ and $stb(\mathcal{D}) = \{\{b, c\}\}$. We omit computation of $uc(\mathcal{D}, c)$ since no contrary is shared among assumptions. Thus $\mathsf{f}_{(\mathbf{E})adstb}(\mathcal{D}, c)$ (forgetting c) is given as $\mathsf{f}_{(\mathbf{E})adstb}(\mathcal{D}, c) = (\mathcal{L}, \mathcal{R}', \mathcal{A}, ^{-})$, where

$$\mathcal{R}' = \{a_c \leftarrow b, c;\ \ b_c \leftarrow a;\ \ c_c \leftarrow c\}.$$

Indeed, this yields $ad(\mathsf{f}_{(\mathbf{E})adstb}(\mathcal{D}, c)) = \{\emptyset, \{a\}\}$ as well as $stb(\mathsf{f}_{(\mathbf{E})adstb}(\mathcal{D}, c)) = \emptyset$ as desired.

The next theorem shows that this was no coincidence and our operator indeed serves our purpose.

Theorem 5.22. *The operator $\mathsf{f}_{(\mathbf{E})adstb}$ is tractable. Moreover, it satisfies $(\mathbf{E})$ for $\sigma \in \{ad, stb\}$.*

Turning to gr semantics, we can simply adapt the idea from Definition 5.14 to obtain a forgetting operator satisfying $(\mathbf{D})$, and $(\mathbf{P})$ that is tractable. Since this is a rather natural modification to our previous operator, we do not discuss this approach in detail here.

5.4 Argument Persistence and Elimination

In this subsection we will focus on the argumentation structure that is induced by the forgetting result. This is closely related to previous work that has been conducted on forgetting in abstract argumentation (Baumann, Gabbay, and Rodrigues 2020; Baumann and Berthold 2022). Recall that argument elimination $(\mathbf{AE})$ requires that all arguments including a are removed and argument persistence $(\mathbf{AP})$ holds if a is removed from all arguments; both notions are thus similar in spirit to elimination $(\mathbf{E})$ and persistence $(\mathbf{P})$.

As usual, we start with a general theoretical analysis before moving on to construct operators.

Theoretical Analysis. We first note that both $(\mathbf{AE})$ and $(\mathbf{AP})$ are satisfiable on their own, but not simultaneously.

Proposition 5.23. *Let $\mathcal{D} = (\mathcal{L}, \mathcal{R}, \mathcal{A}, ^-)$ be an ABAF, $\sigma \in \Sigma$, and $a \in \mathcal{A}$.*

- *There is a forgetting operator* f *satisfying* $(\mathbf{AP})$.
- *There is a forgetting operator* f *satisfying* $(\mathbf{AE})$.
- *There is no operator* f *satisfying both* $(\mathbf{AP})$ *and* $(\mathbf{AE})$.

Next we want to formalize that our instantiation-driven desiderata are not compatible with the semantical ones we discussed above. Indeed, it turns out that in general they are mutually exclusive. That is, there is a trade-off between having intuitive modifications w.r.t. the semantics $\sigma(\mathcal{D})$ and w.r.t. the induced AF $F_{\mathcal{D}}$. We give the following result.

Theorem 5.24. *Let $\mathcal{D} = (\mathcal{L}, \mathcal{R}, \mathcal{A}, ^-)$ be an ABAF and $a \in \mathcal{A}$. For each $\sigma \in \Sigma$, each of the following pairs of desiderata are not simultaneously satisfiable:* $(\mathbf{P})$ *and* $(\mathbf{AP})$*;* $(\mathbf{P})$ *and* $(\mathbf{AE})$*;* $(\mathbf{E})$ *and* $(\mathbf{AP})$*;* $(\mathbf{E})$ *and* $(\mathbf{AE})$.

This result can be shown by building upon known ABA results quite naturally. The underlying reason is that the semantics of an ABAF $\mathcal{D}$ can be computed by means of the corresponding AF $F_{\mathcal{D}}$. Hence, since argument persistence $(\mathbf{AP})$ resp. argument elimination $(\mathbf{AE})$ specify $F_{\mathsf{f}(\mathcal{D},a)}$, these desiderata also uniquely determine the semantics of the ABAF $\mathsf{f}(\mathcal{D}, a)$ after forgetting a.

Lemma 5.25. *Given a forgetting operator* f*, an assumption $a \in \mathcal{A}(\mathcal{D})$ and semantics $\sigma \in \Sigma$.*

- *If* f *satisfies* $(\mathbf{AP})$*, then*
  $\sigma(\mathsf{f}(\mathcal{D}, a)) = \sigma(F_{\mathcal{D}}|_{\{A \setminus \{a\} \vdash s \mid A \vdash s \in A(F_{\mathcal{D}}),\ s \neq a\}})$
- *If* f *satisfies* $(\mathbf{AE})$*, then*
  $\sigma(\mathsf{f}(\mathcal{D}, a)) = \sigma(F_{\mathcal{D}}|_{\{A \vdash s \mid a \notin A\}})$

Here σ both describes ABA as well as AF semantics.

Given this lemma, it suffices to note that the modifications which are required to satisfy the desiderata $(\mathbf{AE})$ and $(\mathbf{AP})$ are not compatible with elimination $(\mathbf{E})$ and persistence $(\mathbf{P})$ as discussed earlier.

Construction of Forgetting Operators. As our last remark on assumption forgetting let us demonstrate how to obtain operators satisfying $(\mathbf{AE})$ resp. $(\mathbf{AP})$. It turns out that the rm operator, which we introduced in Definition 5.9 satisfies argument persistence.

Proposition 5.26. *The operator* $\mathsf{f}_{(\mathbf{AP})}(\mathcal{D}, a) := rm(\mathcal{D}, a)$ *satisfies* $(\mathbf{AP})$.

Example 5.27. *Let $\mathcal{D} = (\mathcal{L}, \mathcal{R}, \mathcal{A}, ^-)$, where*

$$\mathcal{L} = \{a, b, c, p, q, r, s\} \qquad \mathcal{A} = \{a, b, c\}$$
$$\mathcal{R} = \{p \leftarrow a;\ \ q \leftarrow b;\ \ r \leftarrow a, b;\ \ s \leftarrow r, c\}$$

(and some arbitrary contray function). We have arguments $\{a\} \vdash p$, $\{b\} \vdash q$, $\{a, b\} \vdash r$, and $\{a, b, c\} \vdash s$. Applying the remove operator $rm(\mathcal{D}, a)$ yields the set of rules $\mathcal{R}' = \{p \leftarrow;\ \ q \leftarrow b;\ \ r \leftarrow b;\ \ s \leftarrow r, c\}$ with arguments $\top \vdash p$, $\{b\} \vdash q$, $\{b\} \vdash r$, and $\{b, c\} \vdash s$.

We proceed with argument elimination $(\mathbf{AE})$. We did not yet describe a forgetting operator satisfying this property, so we construct a novel one. Recall that argument elimination requires us to remove any argument which relies on the given assumption a. Since we assume our ABAFs to be flat, we do not have to consider cases where a might be entailed from some rule $r \in \mathcal{R}$. It therefore suffices to remove each rule where $a \in body(r)$, thereby ensuring that the arguments making use of such rules cannot be constructed anymore.

Definition 5.28. *Given $\mathcal{D} = (\mathcal{L}, \mathcal{R}, \mathcal{A}, ^-)$ and $a \in \mathcal{A}$ we let $\mathsf{f}_{(\mathbf{AE})}(\mathcal{D}, a) := \big(\mathcal{L} \setminus \{a\}, \mathcal{R}', \mathcal{A} \setminus \{a\}, ^-|_{\mathcal{A} \setminus \{a\}}\big)$ with rules $\mathcal{R}' = \mathcal{R} \setminus \{r \in \mathcal{R} \mid a \in body(r)\}$.*

Example 5.29. *Let $\mathcal{D}$ be the ABAF from Example 5.27. Applying $\mathsf{f}_{(\mathbf{AE})}(\mathcal{D}, a)$ yields $\mathcal{R}' = \{q \leftarrow b;\ \ s \leftarrow r, c\}$ leaving only the argument $\{b\} \vdash q$ constructible.*

Indeed, the operator $\mathsf{f}_{(\mathbf{AE})}(\mathcal{D})$ satisfies argument elimination, as formalized in the following proposition.

Proposition 5.30. *The operator* $\mathsf{f}_{(\mathbf{AE})}(\mathcal{D}, a)$ *satisfies* $(\mathbf{AE})$.

6 On Contrary Forgetting

In this section, we briefly discuss approaches towards forgetting the contrary of an assumption. That is, we interpret contrary forgetting in the sense that defeasibility of the corresponding assumption is forgotten. Of course, strictly technical speaking, this is not possible since by definition each assumption $a \in \mathcal{A}$ has a contrary. However, it is possible to modify $\mathcal{D}$ in a way that $\overline{a}$ does not play any role anymore.

We will see that elimination $(\mathbf{E})$ can be adapted to this setting while we argue that for persistence $(\mathbf{P})$ this might cause counter-intuitive results. As usual we assume that our forgetting operator f is defined for contraries $p \in \overline{\mathcal{A}}$ only.

6.1 Contrary Forgetting Desiderata

To keep this section brief, we will only consider the semantical desiderata here. We need to be cautious, in order to avoid counter-intuitive scenarios. Let us for example consider the primal persistence $(\mathbf{P})$ desideratum we introduced for assumption forgetting. A given forgetting operator f satisfies persistence iff $\sigma(\mathsf{f}(D, a)) = \{E \setminus \{a\} \mid E \in \sigma(D)\}$, i.e. a is removed from each extension. When forgetting the contrary $\overline{a}$ instead of the assumption a, it would therefore seem natural to require $\sigma(\mathsf{f}(D, \overline{a})) = \{E \cup \{a\} \mid E \in \sigma(D)\}$ instead, i.e. a is *added* to each extension. This idea turns out to be too disruptive though, as we illustrate next.

Example 6.1. *Let $\mathcal{D} = (\mathcal{L}, \mathcal{R}, \mathcal{A}, ^-)$, where*

$$\mathcal{L} = \{a, b, a_c, b_c\} \quad \mathcal{R} = \{a_c \leftarrow a;\ \ a_c \leftarrow b;\ \ b_c \leftarrow a\}$$
$$\mathcal{A} = \{a, b\} \qquad\qquad ^- = \{(a, a_c), (b, b_c)\}$$

Considering stable extensions we observe $stb(D) = \{\{b\}\}$. When removing the contrary of a by means of a forgetting operator we expect b to not attack a anymore. However, forgetting $\overline{a}$ has no effect on the rule "$\overline{b} \leftarrow a$" implying that a and b are still incompatible. Thus $\sigma(\mathsf{f}(D, \overline{a})) = \{\{a, b\}\}$ would be the output of some counter-intuitive operator f.

Besides persistence $(\mathbf{P})$ we also considered elimination $(\mathbf{E})$ stating that extensions containing a shall be removed. This is a more cautious approach since it does not rely on modifying the interactions of assumptions, but merely removes some models of the knowledge base. We consider the following "contrary forgetting" version of elimination.

Desideratum 6.2. *Given an ABAF $\mathcal{D}$ and an assumption $a \in \mathcal{A}$. A forgetting operator* f *satisfies*

$(\mathbf{CE})$ contrary elimination *iff*

$$\sigma(\mathsf{f}(\mathcal{D}, \overline{a})) = \begin{cases} \{E \in \sigma(\mathcal{D}) \mid a \in E\} & \text{if } a \in \bigcup \sigma(\mathcal{D}), \\ \{\emptyset\} & \text{if } \sigma \neq stb, a \notin \bigcup \sigma(\mathcal{D}), \\ \emptyset & \text{if } \sigma = stb, a \notin \bigcup \sigma(\mathcal{D}). \end{cases}$$

Hence, the forgetting operator f shall keep only extensions containing a. In corner cases $\emptyset$ is allowed.

Example 6.3. *Recall the running example about planning the vacation. Removing the outcomes in which Antoine is not joining the trip corresponds to forgetting about $antoine\_joins_c$ via* $(\mathbf{CE})$.

6.2 Contrary Elimination Forgetting

The attentive reader may already expect that $(\mathbf{CE})$ is not satisfiable for all semantics. As usual, for the positive cases we provide corresponding forgetting operators.

Theoretical Analysis. If $a \in \mathcal{A}$ is credulously accepted w.r.t. ad, then a forgetting operator f satisfying $(\mathbf{CE})$ would output an ABAF s.t. $\emptyset \notin ad(\mathsf{f}(\mathcal{D}, \overline{a}))$; this is of course impossible since the empty set is always admissible. We therefore conclude that no such operator exists. Regarding complete semantics, consider the following example.

Example 6.4. *Let $\mathcal{D} = (\mathcal{L}, \mathcal{R}, \mathcal{A}, ^-)$, s.t. $\mathcal{A} = \{a, b, c, d\}$, $\mathcal{L} = \mathcal{A} \cup \{a_c, b_c, c_c, d_c\}$, $\overline{a} = a_c$ for all $a \in \mathcal{A}$, and $\mathcal{R}$:*

$$c_c \leftarrow b \qquad b_c \leftarrow c \qquad a_c \leftarrow d \qquad d_c \leftarrow b \qquad d_c \leftarrow c$$

That is, b and c mutually attack each other, both defending a from the attack coming from d. We therefore conclude $co(\mathcal{D}) = \{\emptyset, \{a, b\}, \{a, c\}\}$. If we want to forget a_c satisfying $(\mathbf{CE})$, then we require $\sigma(\mathsf{f}(\mathcal{D}, a_c)) = \{\{a, b\}, \{a, c\}\}$ which is impossible due to the lack of a unique grounded extension.

For the remaining semantics gr, pr, and stb we do not face such issues. We infer the following possibility result regarding contrary elimination $(\mathbf{CE})$.

Theorem 6.5. *Let $\mathcal{D} = (\mathcal{L}, \mathcal{R}, \mathcal{A}, ^-)$ be an ABAF, $\sigma \in \Sigma$, and $a \in \mathcal{A}$. There is a forgetting operator* f *satisfying* $(\mathbf{CE})$ *iff $\sigma \in \{gr, pr, stb\}$.*

Notably, we do not require any complexity-theoretic argument to further distinguish between the semantics. Indeed, we find tractable forgetting operators in all three cases.

Construction of Forgetting Operators. Satisfaction of $(\mathbf{CE})$ relies on imposing constraints on the given set of extensions, as it was the case for the elimination notion $(\mathbf{E})$ for assumption forgetting. This time we introduce a fresh assumption x which attacks itself as well as each assumption in $\mathcal{D}$. Then, we construct an attack from a to x. This way, stable resp. preferred extensions containing a remain unaffected, but the remaining ones get excluded.

Definition 6.6. *Given $\mathcal{D} = (\mathcal{L}, \mathcal{R}, \mathcal{A}, ^-)$ and $a \in \mathcal{A}$ we let $\mathsf{f}_{(\mathbf{CE})prstb}(\mathcal{D}, \overline{a}) = (\mathcal{A} \cup \{x\}, \mathcal{R}', \mathcal{L} \cup \{x, x_c\}, ^- \cup (x, x_c))$ where $\mathcal{R}' = \mathcal{R} \cup \{\overline{b} \leftarrow x \mid b \in \mathcal{A} \cup \{x\}\} \cup \{x_c \leftarrow a\}$.*

We formalize that $\mathsf{f}_{(\mathbf{CE})prstb}(\mathcal{D}, \overline{a})$ is indeed suitable.

Theorem 6.7. *The operator $\mathsf{f}_{(\mathbf{CE})prstb}$ satisfies contrary-elimination for $\sigma \in \{pr, stb\}$.*

We note that we can construct an operator for grounded semantics that in addition satisfies deletion $(\mathbf{D})$, by using similar methods as in the case of assumption persistence.

7 Conclusion

In this paper, we covered several forgetting aspects in assumption-based argumentation (ABA). We adapted common desiderata from the literature to our setting and comprehensively investigated their (un)satisfiability. We considered forgetting ordinary atoms, assumptions, and contraries, both semantical as well as syntactical properties of our forgetting operators, but also their impact on the instantiation procedure. Whenever possible we constructed a suitable polynomial-time computable operator.

We want to mention that forgetting contraries as a dual version of assumptions forgetting is similar in spirit to enforcement. While similar problems have been studied in the context of formal argumentation (Baumann and Brewka 2010; Baumann and Brewka 2015; Borg and Bex 2021; Rapberger and Ulbricht 2022), we are not aware of any work in the context of ABA which studies this particular notion.

An interesting future work direction would be the consideration of more natural, non-trivial forgetting operators. For example our study did not cover cases where in general there is no forgetting operator. However, it might be possible to find operators that work in most cases, for certain ABA fragments, or approximate the target set of extensions. Moreover, we only covered a handful of desiderata, all of which are quite strong. A more comprehensive study of further ones or weaker versions of those we proposed would be an interesting topic for future research as well. More broadly, further forgetting research in ABA would benefit from a thorough investigation of its expressive power, as it has been conducted for AFs (Baumann and Strass 2013; Dunne et al. 2015; Ulbricht 2021). We mention that forgetting has already been applied successfully to shorten proofs in description logics (Alrabbaa et al. 2020). It would be compelling to investigate whether similar results can be achieved for ABA. Finally, this paper focused on ABA as the underlying formalism of the structured arguments. Considering forgetting in other well-established argumentation formalisms, like defeasible logic programming (Moguillansky et al. 2008), may be an interesting avenue for future studies.

Acknowledgements

This research has been supported by the Vienna Science and Technology Fund (WWTF) through project ICT19-065, the Austrian Science Fund (FWF) through project P32830, and by the Federal Ministry of Education and Research of Germany and by Sächsische Staatsministerium für Wissenschaft, Kultur und Tourismus in the programme Center of Excellence for AI-research "Center for Scalable Data Analytics and Artificial Intelligence Dresden/Leipzig", project identification number: ScaDS.AI.
This work was partially conducted while Matti was a visiting PhD student at the University of Luxembourg.

References

Alrabbaa, C.; Baader, F.; Borgwardt, S.; Koopmann, P.; and Kovtunova, A. 2020. Finding small proofs for description logic entailments: Theory and practice (extended technical report). *arXiv preprint arXiv:2004.08311*.

Baumann, R., and Berthold, M. 2022. Limits and possibilities of forgetting in abstract argumentation. In *Proceedings of (IJCAI-22)*, 2539–2545.

Baumann, R., and Brewka, G. 2010. Expanding argumentation frameworks: Enforcing and monotonicity results. In *Proceedings of (COMMA-10)*, 75–86.

Baumann, R., and Brewka, G. 2015. AGM meets abstract argumentation: Expansion and revision for dung frameworks. In *Proceedings of (IJCAI-15)*, 2734–2740.

Baumann, R., and Strass, H. 2013. On the maximal and average numbers of stable extensions. In *Proceedings of (TAFA-13)*, 111–126.

Baumann, R., and Ulbricht, M. 2019. If nothing is accepted–repairing argumentation frameworks. *Journal of Artificial Intelligence Research* 1099–1145.

Baumann, R.; Gabbay, D. M.; and Rodrigues, O. 2020. Forgetting an argument. In *Proceedings of (AAAI-20)*, 2750–2757.

Berthold, M.; Gonçalves, R.; Knorr, M.; and Leite, J. 2019. A syntactic operator for forgetting that satisfies strong persistence. *Theory and Practice of Logic Programming* 1038–1055.

Berthold, M. 2022. On syntactic forgetting with strong persistence. In *Proceedings of (KR-22)*.

Borg, A., and Bex, F. 2021. Enforcing sets of formulas in structured argumentation. In *Proceedings of (KR-21)*, 130–140.

Čyras, K.; Fan, X.; Schulz, C.; and Toni, F. 2018. Assumption-based argumentation: Disputes, explanations, preferences. In *Handbook of Formal Argumentation*. chapter 7, 365–408.

Delgrande, J. P., and Wang, K. 2015. A syntax-independent approach to forgetting in disjunctive logic programs. In *Proceedings of (AAAI-15)*, 1482–1488.

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

Dunne, P. E.; Dvořák, W.; Linsbichler, T.; and Woltran, S. 2015. Characteristics of multiple viewpoints in abstract argumentation. *Artificial Intelligence* 153–178.

Dvorák, W., and Dunne, P. E. 2018. Computational problems in formal argumentation and their complexity. In *Handbook of Formal Argumentation*.

Eiter, T., and Wang, K. 2008. Semantic forgetting in answer set programming. *Artificial Intelligence* 1644–1672.

Falappa, M. A.; Kern-Isberner, G.; and Simari, G. R. 2009. Belief revision and argumentation theory. In *Argumentation in Artificial Intelligence*. 341–360.

Gabbay, D.; Giacomin, M.; Simari, G. R.; and Thimm, M., eds. 2021. *Handbook of Formal Argumentation*.

Gonçalves, R.; Janhunen, T.; Knorr, M.; and Leite, J. 2021. On syntactic forgetting under uniform equivalence. In *Proceedings of (JELIA-21)*, 297–312.

Gonçalves, R.; Knorr, M.; and Leite, J. 2016. You can't always forget what you want: On the limits of forgetting in answer set programming. In *Proceedings of (ECAI-16)*, 957–965.

Haret, A.; Wallner, J. P.; and Woltran, S. 2018. Two sides of the same coin: Belief revision and enforcing arguments. In *Proceedings of (IJCAI-18)*, 1854–1860.

Knorr, M., and Alferes, J. J. 2014. Preserving strong equivalence while forgetting. In *Proceedings of (JELIA-14)*, 412–425.

Lin, F., and Reiter, R. 1994. Forget it. In *Working Notes of AAAI Fall Symposium on Relevance*, 154–159.

Moguillansky, M. O.; Rotstein, N. D.; Falappa, M. A.; García, A. J.; and Simari, G. R. 2008. Argument theory change applied to defeasible logic programming. In *Proceedings of (AAAI-08)*, 132–137.

Niskanen, A.; Wallner, J. P.; and Järvisalo, M. 2018. Extension enforcement under grounded semantics in abstract argumentation. In *Proceedings of (KR-18)*, 178–183.

Oikarinen, E., and Woltran, S. 2011. Characterizing strong equivalence for argumentation frameworks. *Artificial Intelligence* 1985–2009.

Prakken, H. 2022. Formalising an aspect of argument strength: Degrees of attackability. In *Proceedings of (COMMA-22)*, 296–307.

Rapberger, A., and Ulbricht, M. 2022. On dynamics in structured argumentation formalisms. In *Proceedings of (KR-22)*.

Toni, F. 2014. A tutorial on assumption-based argumentation. *Argument and Computation* 89–117.

Ulbricht, M. 2021. On the maximal number of complete extensions in abstract argumentation frameworks. In *Proceedings of (KR-21)*, 707–711.

Wallner, J. P.; Niskanen, A.; and Järvisalo, M. 2016. Complexity results and algorithms for extension enforcement in abstract argumentation. In *Proceedings of (AAAI-17)*, 1088–1094.

Wallner, J. P. 2020. Structural constraints for dynamic operators in abstract argumentation. *Argument and Computation* 151–190.

Wang, Y.; Zhang, Y.; Zhou, Y.; and Zhang, M. 2014. Knowledge forgetting in answer set programming. *Journal of Artificial Intelligence Research* 31–70.

Wang, Y.; Wang, K.; and Zhang, M. 2013. Forgetting for answer set programs revisited. In *Proceedings of (IJCAI-13)*, 1162–1168.

Zhang, Y., and Foo, N. Y. 2006. Solving logic program conflict through strong and weak forgettings. *Artificial Intelligence* 739–778.

Zhang, Y., and Zhou, Y. 2009. Knowledge forgetting: Properties and applications. *Artificial Intelligence* 1525–1537.