Computational Models of Argument
F. Toni et al. (Eds.)
© 2022 The authors and IOS Press.
This article is published online with Open Access by IOS Press and distributed under the terms of the Creative Commons Attribution Non-Commercial License 4.0 (CC BY-NC 4.0).
doi:10.3233/FAIA220154

Just a Matter of Perspective

Intertranslating Expressive Argumentation Formalisms

Matthias KÖNIG [a], Anna RAPBERGER [a], and Markus ULBRICHT [b]
[a] *TU Wien, Institute of Logic and Computation*
[b] *Leipzig University, Department of Computer Science*

Abstract. Many structured argumentation approaches proceed by constructing a Dung-style argumentation framework (AF) corresponding to a given knowledge base. While a main strength of AFs is their simplicity, instantiating a knowledge base oftentimes requires exponentially many arguments or additional functions in order to establish the connection. In this paper we make use of more expressive argumentation formalisms. We provide several novel translations by utilizing claim-augmented AFs (CAFs) and AFs with collective attacks (SETAFs). We use these frameworks to translate assumption-based argumentation (ABA) frameworks as well as logic programs (LPs) into the realm of graph-based argumentation.

Keywords. Argumentation, Translations, ABA, CAF, Logic Programs, SETAF

1. Introduction

Argumentation structures often arise from instantiating knowledge bases and identifying their relevant conflicts. The representation of knowledge bases in terms of graph-based argumentation formalisms has several advantages. First, they provide an intuitive and user-friendly way for conflict-representation due to their graphical design. Second, the uniform representation allows to compare different, seemingly unrelated knowledge bases and helps to identify their similarities. Various kinds of knowledge bases and applications lead to the invention of several tailor-made argumentation formalisms, each with their own advantages and disadvantages. In formal argumentation, *Abstract Argumentation* due to Dung [1] serves as a common denominator for many of these formalisms. Popular extensions of Dung's original framework incorporate for example propositional acceptance conditions [2], assumptions [3], claims [4], or collective attacks [5]. At first glance, these formalisms seem incompatible due to their focus on seemingly entirely different features. In an effort to relate selected formalisms, researchers singled out pairs of formalisms and provided translations for the respective cases. For the classical Dung semantics, i.e., for complete, preferred, stable, and grounded semantics ($com, pref, stb, grd$), semantics-preserving translations have been successfully established in many cases.

In this work, we take a step back and compare a variety of argumentation formalisms, namely Assumption Based Argumentation (ABA) [3], Claim-Augmented Frameworks (CAF) [4], and Argumentation Frameworks with Collective Attacks (SETAF) [5]. Moreover we consider the closely related Normal Logic Programs (LP) and the restricted atomic LPs [6] (we expect readers to enjoy this work the most if they are al-

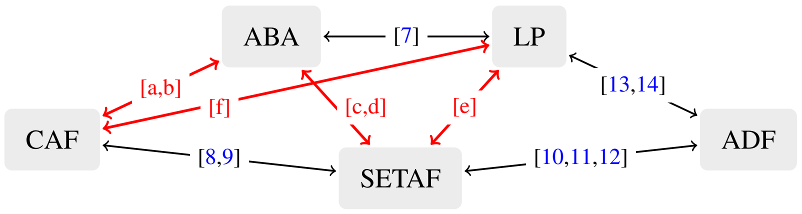


Figure 1. Overview of existing and novel transformations. Novel translations between ABA and CAFs are given in [a] Def. 3.6 and [b] 3.9; we present two translations relating ABA and SETAFs, cf. [c] Def. 3.13 and [d] 3.17; translations between [e] SETAFs and LP are in Section 4.2; and for [f] CAFs and LPs by Def. 4.3.

ready familiar with some of these formalisms). There already exist semantics-preserving translations between several classes of the aforementioned formalisms. Caminada and Schulz [7] provide a translation between ABA and LP and vice versa. In [8,9], the correspondence between well-formed CAFs and SETAFs has been settled. All of these mentioned translations preserve complete, stable, and preferred models (extensions).

If we furthermore take the well-investigated relation between Abstract Dialectical Frameworks (ADF) [2] and LPs [13,14] as well as to SETAFs, respectively [10,11,12], into account and collect all available results, we obtain the following insight: (classes of) ABA frameworks, LPs, ADFs, SETAFs, and CAFs can all be viewed, to some extent, as different sides of the same (pentagonal) coin. We summarize this insight in Figure 1. We note that not all translations consider all instances of the domain; e.g., the translation from CAFs to SETAFs restricts to so-called well-formed CAFs; also, Dvořák et. al [10] as well as Alcântara and Sá [11] focus on attacking (support-free) ADFs. Likewise, the *image* of the translation often do not cover all instances of the target formalism, e.g., Polberg [12] translates SETAFs into attacking ADFs and Caminada and Schulz [7] map LPs to a sub-class of ABA frameworks. As one can verify by following the directed arrows, there exists semantics-preserving rewriting methods between (classes of) all of these formalisms. While this existential statement suffices to establish a theoretical correspondence it is hardly of practical use for translating, e.g., ABA instances to CAFs (this concrete example would require the application of four different translations). From a theoretical point of view, one would have to comprehend several steps through various different formalisms, thereby missing the observation that there are immediate translations which preserve the structure quite well, as we will establish in this paper. For example the CAF obtained from an ABA framework is natural and can be constructed directly, and the role of the additional claims becomes clear immediately.

The paper is organized as follows. In Section 3 we focus on the intertranslatability of ABA, CAFs, and SETAFs. We show how an ABA framework naturally induces a CAF which preserves the structure of the knowledge base due to the flexible handling of claims. Moreover, we explore the advantageous features of SETAFs which yield a representation that requires fewer arguments. We will show that if one is solely interested in the underlying assumptions, SETAFs yield impressively concise representations. In Section 4 we discuss the close relation between atomic LPs, CAFs, and SETAFs, provide natural pairwise translations and demonstrate their compatibility. Along the way, we show that the instantiation procedure [15] (i.e. constructing arguments from a general LPs) can be bridged by first making the LP atomic.

We omit proofs in the present paper; full proofs are made available at https://www.dbai.tuwien.ac.at/research/report/dbai-tr-2022-123.pdf.

2. Background

We recall the necessary background for AFs since they constitute our main underlying formalism. The other formalisms will be introduced on the fly. An argumentation framework (AF) [1] is a directed graph (A,R) where A is a finite set of arguments and $R \subseteq A \times A$ the attack relation. An argument x (set $E \subseteq A$) *attacks* y if $(x,y) \in R$ (some $z \in E$ attacks y). We write $E_R^+ = \{a \in A \mid E \text{ attacks } a\}$ and $E_R^- = \{a \in A \mid (a,b) \in R, b \in E\}$, and for short $x_R^+ = \{x\}_R^+$, $x_R^- = \{x\}_R^-$; we omit subscript R if it is clear from the context.

A set $E \subseteq A$ is *conflict-free* in $F = (A,R)$ iff $(x,y) \notin R$ for all $x,y \in E$; E *defends* an argument x if E attacks each attacker of x. A conflict-free set E is *admissible* in F ($E \in adm(F)$) iff it defends all its elements. A *semantics* σ is a function which returns a set of subsets of A. These subsets are called σ*-extensions*. In this paper we consider so-called *complete*, *grounded*, *preferred*, and *stable* semantics (abbr. *com*, *grd*, *pref*, *stb*).

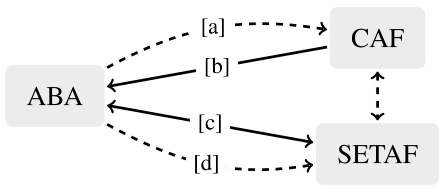
Definition 2.1. Let $F = (A,R)$ be an AF and $E \in adm(F)$. We let $E \in com(F)$ iff E contains all arguments it defends; $E \in grd(F)$ iff E is $\subseteq$-minimal in $com(F)$; $E \in pref(F)$ iff E is $\subseteq$-maximal in $com(F)$; $E \in stb(F)$ iff $E^+ = A \setminus E$.

Throughout the paper we will frequently use the notion of a hitting set: Let $\mathscr{M}$ be a set of sets. We call $\mathscr{H}$ a *hitting set* of $\mathscr{M}$ if $\mathscr{H} \cap M \neq \emptyset$ for each $M \in \mathscr{M}$. By $HS_{min}(\mathscr{M})$ we denote the $\subseteq$-minimal hitting sets of $\mathscr{M}$. We will make use of the following result.

Lemma 2.2 ([16]). *Let $X = \{X_1, \ldots, X_n\}$ be a set of sets with $X_i \not\subseteq X_j$ for $i \neq j$. Then $HS_{min}(HS_{min}(X)) = X$.*

3. Intertranslatability of ABA Frameworks, CAFs, and SETAFs

In this section, we consider the relation between ABA frameworks, well-formed CAFs, and SETAFs. Semantics for ABA can be equivalently formulated in terms of *assumptions* or in terms of *arguments* via attacks based on their *claims*. There are different representations that put the focus on either preserving assumption-sets or extensions in terms of conclusions. Figure 2 shows the different translations and directions we consider in this section: while the CAF representation focuses on extensions in terms of conclusions but also preserves assumption-extension under projection (cf. translation [a] in Figure 2), there are several possibilities to represent ABA frameworks as SETAFs. Translation [c] relates assumptions in the ABA framework with arguments in the SETAF while Translation [d] relates conclusions with arguments. We also consider the reversed direction, i.e., constructing ABA frameworks from CAFs and SETAFs (cf. [b] and [c], respectively). In Section 3.1, we consider the relation of ABA and CAFs; in Section 3.2 we examine the relation between ABA and SETAFs. First, we provide necessary background for ABA.



Translations [a,d] from ABA to CAFs and SETAFs preserve conclusions (cf. Def. 3.6 and 3.17); Translation [b] from CAFs to ABA preserves proper conclusion-extensions (cf. Def. 3.9); Translation [c] between ABA and SETAFs preserves assumption-sets (cf. Def. 3.13). The diagram commutes w.r.t. dashed lines (cf. Prop. 3.21).

Figure 2. Semantics-preserving translations between ABA frameworks, CAFs, and SETAFs.

Assumption-based Argumentation. We assume a deductive system $(\mathcal{L},\mathcal{R})$, where $\mathcal{L}$ is a formal language and $\mathcal{R}$ is a set of inference rules of the form $r : a_0 \leftarrow a_1,\dots,a_n$, $a_i \in \mathcal{L}$; $head(r) = a_0$ denotes the head and $body(r) = \{a_1,\dots,a_n\}$ the body of rule r.

Definition 3.1. An ABA framework is a tuple $(\mathcal{L},\mathcal{R},\mathcal{A},^{-})$, where $(\mathcal{L},\mathcal{R})$ is a deductive system, $\mathcal{A} \subseteq \mathcal{L}$, $\mathcal{A} \neq \emptyset$ a set of assumptions, and a contrary function $^{-} : \mathcal{A} \to \mathcal{L}$.

We focus on ABA frameworks which are *flat*, i.e., for each rule $r \in \mathcal{R}$, $head(r) \notin \mathcal{A}$, and *finite*, i.e., $\mathcal{L}$, $\mathcal{R}$, $\mathcal{A}$ are finite. Furthermore, we assume $\mathcal{L}$ to be a set of atoms.

An atom $p \in \mathcal{L}$ in an ABA framework $D = (\mathcal{L},\mathcal{R},\mathcal{A},^{-})$ is *tree-derivable* from assumptions $S \subseteq \mathcal{A}$ and rules $R \subseteq \mathcal{R}$, denoted by $S \vdash_R p$, if there is a finite rooted labeled tree such that the root is labeled with p, the set of labels for the leaves is equal to S or $S \cup \{\top\}$, and there is a surjective mapping from the set of internal nodes to R s.t. each internal note v is labeled with $head(r)$ for some $r \in R$ and the set of all successor nodes corresponds to $body(r)$ or $\top$ if $body(r) = \emptyset$. We write $S \vdash p$ if there exists $R \subseteq \mathcal{R}$ with $S \vdash_R p$. Derivability for a set of assumptions $S \subseteq \mathcal{A}$ is defined via $Th_D(S) = \{p \mid S \vdash p\}$.

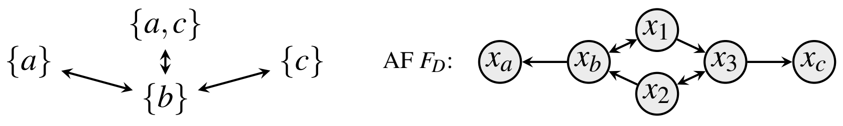
A set $S \subseteq \mathcal{A}$ *attacks* $a \in \mathcal{A}$ if there is $S' \subseteq S$ such that $S' \vdash \overline{a}$; S attacks $T \subseteq \mathcal{A}$ if it attacks some $a \in T$. S is conflict-free if it does not attack itself; S is admissible if it is conflict-free and counter-attacks each attacker (we say: S defends itself). We recall grounded, complete, preferred, and stable ABA semantics (abbr. *grd*, *com*, *pref*, *stb*).

Definition 3.2. For an ABA $D = (\mathcal{L},\mathcal{R},\mathcal{A},^{-})$ and an admissible set $S \subseteq \mathcal{A}$, $S \in com(D)$ iff S contains every assumption it defends; $S \in grd(D)$ iff S is $\subseteq$-minimal in $com(D)$; $S \in pref(D)$ iff S is $\subseteq$-maximal in $com(D)$; $S \in stb(D)$ iff S attacks each $\{x\} \subseteq \mathcal{A} \setminus S$. Given $\sigma \in \{com, grd, pref, stb\}$, the *$\sigma$-conclusion-extensions* of D are $\sigma_{Th}(D) = \{Th_D(S) \mid S \in \sigma(D)\}$, the *proper* σ-conclusion-extensions of D are given by $\{C \setminus \mathcal{A} \mid C \in \sigma_{Th}(D)\}$.

ABA frameworks and AFs are closely related (see, e.g., [17]). Viewing tree derivations as arguments, an ABA framework induces a corresponding AF as follows.

Definition 3.3. The associated AF $F_D = (A,R)$ of an ABA $D = (\mathcal{L},\mathcal{R},\mathcal{A},^{-})$ is given by $A = \{S \vdash p \mid \exists R \subseteq \mathcal{R} : S \vdash_R p\}$ and attack relation $(S_1 \vdash p, S_2 \vdash q) \in R$ iff $p \in \{\overline{s} \mid s \in S_2\}$.

Example 3.4. Consider the ABA D with assumptions $\mathcal{A} = \{a,b,c\}$ and rules $r_1 : p \leftarrow a$, $r_2 : p \leftarrow c$, and $r_3 : q \leftarrow b$. Moreover, $\overline{a} = b$, $\overline{b} = p$, and $\overline{c} = q$. Below we depict the attacks between the assumption-sets (left, we omit $\emptyset$, $\{a,b\}$, $\{b,c\}$, and $\mathcal{A}$) and the AF F_D (right) with arguments x_i (induced by rules r_i) and arguments x_a, x_b, x_c for the assumptions.



The ABA D has two stable assumption-sets: $S_1 = \{b\}$ and $S_2 = \{a,c\}$ with $Th_D(S_1) = \{b,q\}$ and $Th_D(S_2) = \{a,c,p\}$. The stable extensions in F_D are $\{x_3,x_b\}$ and $\{x_1,x_2,x_a,x_c\}$.

For an argument $x = S \vdash p$, we consider functions $cl(x) = p$ and $asms(x) = S$; moreover, $cl(E) = \{cl(x) \mid x \in E\}$ and $asms(E) = \bigcup_{x \in E} asms(x)$ for a set of arguments E.

Proposition 3.5 ([17]). *For an ABA D, its associated AF F, $\sigma \in \{grd, com, pref, stb\}$; if $E \in \sigma(F)$ then $asms(E) \in \sigma(D)$; and if $S \in \sigma(D)$ then $\{S' \vdash p \mid \exists S' \subseteq S : S' \vdash p\} \in \sigma(F)$.*

3.1. Assumption-based Argumentation and Claims

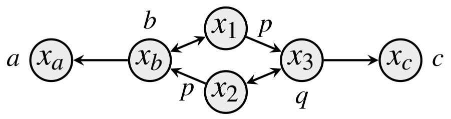
Claim-augmented Argumentation Frameworks. A *claim-augmented argumentation framework (CAF)* [4] is a triple $\mathcal{F} = (A,R,cl)$ where $F = (A,R)$ is an AF and a function cl which assigns a claim to each argument in A. The claim-function is extended to sets in the natural way, i.e. for a set $E \subseteq A$, we let $cl(E) = \{cl(a) \mid a \in E\}$. For a CAF $\mathcal{F} = (A,R,cl)$, $F = (A,R)$, and an AF semantics σ, we define $\sigma_c(\mathcal{F}) = \{cl(E) \mid E \in \sigma(F)\}$. In this work, we focus on CAFs that are *well-formed*; i.e. CAFs satisfying $a_R^+ = b_R^+$ for all $a,b \in A$ with $cl(a) = cl(b)$. Whenever we write CAF, we mean well-formed CAF.

ABA-CAF Translations. There is a natural adaption of the AF instantiation given in Definition 3.3 to CAFs by assigning each argument $S \vdash p$ its claim p:

Definition 3.6. The associated CAF $\mathcal{F}_D = (A,R,cl)$ for an ABA $D = (\mathcal{L},\mathcal{R},\mathcal{A},\overline{})$ is obtained by constructing (A,R) from Definition 3.3 and $cl(S \vdash p) = p$ for all $S \vdash p \in A$.

Example 3.7. Instantiating ABA D from Example 3.4 yields the following CAF:

CAF $\mathcal{F}_D$:



$$\begin{tikzcd} & & x_1 \arrow[dr] \arrow[dl, leftrightarrow] & & \\ x_a & x_b \arrow[l] & & x_3 \arrow[r] & x_c \\ & & x_2 \arrow[ul] \arrow[ur, leftrightarrow] & & \end{tikzcd}$$

The CAF $\mathcal{F}_D$ is well-formed since attacks depend on the conclusion of the attacking argument: an argument x attacks argument y if $cl(x) = \overline{a}$ for some $a \in asms(y)$. Due to Proposition 3.5, the translation preserves the σ-conclusion-extensions of an ABA D; assumption-extensions can be obtained by restricting the conclusion-sets to $\mathcal{A}$.

Proposition 3.8. *For an ABA $D = (\mathcal{L},\mathcal{R},\mathcal{A},\overline{})$, its associated CAF $\mathcal{F}_D$ and $\sigma \in \{grd, com, pref, stb\}$, it holds that $\sigma_{Th}(D) = \sigma_c(\mathcal{F}_D)$ and $\sigma(D) = \{C \cap \mathcal{A} \mid C \in \sigma_c(\mathcal{F}_D)\}$.*

For the other direction, we identify each claim c in a given well-formed CAF as contrary of some hidden assumption a_c; moreover, each argument which is attacked by claim c is derived from assumption a_c (i.e., a_c is attacked by all arguments with claim c).

Definition 3.9. The associated ABA $D_{\mathcal{F}} = (\mathcal{L},\mathcal{R},\mathcal{A},\overline{})$ of a CAF $\mathcal{F} = (A,R,cl)$ is given by $\mathcal{A} = \{a_c \mid c \in cl(A)\}$, $\mathcal{L} = \mathcal{A} \cup cl(A)$, contrary function $\overline{a_c} = c$ for all $c \in cl(A)$, and $\mathcal{R} = \{cl(x) \leftarrow \{a_{cl(y)} \mid y \in x^-\} \mid x \in A\}$.

We obtain a translation which relates claim-sets of the CAF with the *proper* conclusion-extensions of the obtained ABA. Note the restriction to the proper conclusion-extensions is necessary since the translation treats assumptions as implicit information.

Proposition 3.10. *For a CAF $\mathcal{F} = (A,R,cl)$, its corresponding ABA $D_{\mathcal{F}}$ and a semantics $\sigma \in \{grd, com, pref, stb\}$, it holds that $\sigma_c(\mathcal{F}) = \{C \setminus \mathcal{A} \mid C \in \sigma_{Th}(D_{\mathcal{F}})\}$.*

Example 3.11. Consider the CAF $\mathcal{F}_D$ from Example 3.7. We construct an ABA $D_{\mathcal{F}_D} = (\mathcal{L},\mathcal{R},\mathcal{A},\overline{})$ with $\mathcal{A} = \{a_p, a_q, a_a, a_b, a_c\}$, contrary function $\overline{a_x} = x$ for each claim in $\mathcal{F}_D$ and rules $p \leftarrow a_b$, $p \leftarrow a_q$, $q \leftarrow a_p$, $a \leftarrow a_b$, $b \leftarrow a_p$, and $c \leftarrow a_q$. The ABA $D_{\mathcal{F}_D}$ has two stable assumption-sets $S_1 = \{a_a, a_p, a_c\}$ and $S_2 = \{a_b, a_q\}$ with $Th_{D_{\mathcal{F}_D}}(S_1) = \{a_a, a_p, a_c, b, q\}$ and $Th_{D_{\mathcal{F}_D}}(S_2) = \{a_b, a_q, a, c, p\}$. The proper conclusion-extensions of $D_{\mathcal{F}_D}$ are $\{b,q\}$ and $\{a,c,p\}$ which correspond to the conclusion-extensions of D.

3.2. Assumption-based Argumentation and Collective Attacks

Argumentation frameworks with collective attacks. A SETAF [18] is a pair $SF = (A,R)$ where A is a finite set of arguments and $R \subseteq (2^A \setminus \{\emptyset\}) \times A$ is the attack relation. For an attack $(T,h) \in R$ we call T the *tail* and h the *head* of the attack. SETAFs (A,R) where $|T| = 1$ for all $(T,h) \in R$ amount to AFs. In that case, we write (t,h) to denote $(\{t\},h)$.

A set $T_1 \subseteq A$ attacks $h \in A$ (the set $T_2 \subseteq A$) if there is $T_1' \subseteq T_1$ (and $h \in T_2$, resp.) such that $(T_1',h) \in R$. We write $h_R^- = \{T \mid (T,h) \in R\}$ to denote the set of attackers of the argument h (in R). For $S \subseteq A$, we use S_R^+ to denote the set of arguments attacked by S (in R). S is *conflict-free* in SF if it does not attack itself; S defends argument $a \in A$ if it attacks each attacker of a; likewise, S defends $T \subseteq A$ iff it defends each $a \in T$. A set S is called *admissible* if it defends itself ($adm(SF)$ denotes the set of all admissible sets in SF). AF semantics generalize to SETAFs in the following way [19,5].

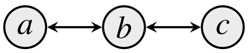
Definition 3.12. Given a SETAF $SF = (A,R)$ and a set $S \in adm(SF)$. Then, $S \in com(SF)$ iff S contains each argument it defends; $S \in grd(SF)$ iff S is $\subseteq$-minimal in $com(SF)$; $S \in pref(SF)$ iff S is $\subseteq$-maximal in $com(SF)$; $S \in stb(SF)$ iff S attacks all $a \in A \setminus S$.

ABA-SETAF-translations: relating assumptions with arguments. When inspecting the definitions of attacks for ABA frameworks and SETAFs we find the following natural correspondence: a set of arguments T attacks an argument h in the SETAF iff T derives the contrary of h in the corresponding ABA. We obtain an ABA framework from a given SETAF by introducing a rule $\overline{h} \leftarrow T$ for each attack $(T,h) \in R$. For the other direction, we identify conflicts between assumption-sets. Below, we give the resulting translations.

Definition 3.13. For an ABA $D = (\mathcal{L},\mathcal{R},\mathcal{A},\overline{\ })$, we define the corresponding SETAF $SF_D = (A_D,R_D)$ with $A_D = \mathcal{A}$ and $(S,a) \in R_D$ iff $S \vdash \overline{a}$. For a SETAF $SF = (A,R)$, we define the corresponding ABA $D_{SF} = (\mathcal{L}_{SF},\mathcal{R}_{SF},\mathcal{A}_{SF},\overline{\ })$ with $\mathcal{L}_{SF} = A \cup \{p_x \mid x \in A\}$, $\mathcal{A}_{SF} = A$, $\overline{x} = p_x$ for all $x \in A$, and for each $(T,h) \in R$, we add a rule $p_h \leftarrow T$ to $\mathcal{R}_{SF}$.

Example 3.14. Instantiating ABA D from Example 3.4 yields the following SETAF:

SETAF SF_D:



The translations indeed preserve the (assumption-based) semantics.

Proposition 3.15. *Given a semantics $\sigma \in \{grd, com, pref, stb\}$. For an ABA D and its associated SETAF SF_D, it holds that $\sigma(D) = \sigma(SF_D)$. For a SETAF SF and its associated ABA D_{SF}, it holds that $\sigma(SF) = \sigma(D_{SF})$.*

We obtain the following strong intertranslatibility result using the correspondence $(S,a) \in R$ in SF iff $\overline{a} \leftarrow S$ in D_{SF} iff $S \vdash \overline{a}$ in D_{SF} iff $(S,a) \in R$ in $SF_{D_{SF}} = SF$.

Proposition 3.16. *Given a SETAF SF, it holds that $SF_{D_{SF}} = SF$.*

This result shows that no information is lost in the SETAF when representing it in terms of ABA. The other direction, i.e., translating ABA frameworks to SETAFs, however, comes with a cost: given an ABA framework D, it is impossible to extract the σ-conclusion-extensions from SETAF SF_D. This means that the conclusions of a given ABA instance are lost when applying the translation. In the following, we present a translation that preserves conclusions of an ABA instance.

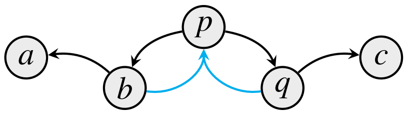
ABA-SETAF-translations: relating conclusions and arguments. In order to establish a translation from ABA frameworks to SETAFs that preserves the conclusions of the original instance, we proceed as follows: For a given ABA instance $D = (\mathcal{L}, \mathcal{R}, \mathcal{A}, ^{-})$, we construct a corresponding SETAF $SF_D = (A, R)$ with

1. $A = \{p \mid \exists S \subseteq \mathcal{A} : S \vdash p\}$, i.e., conclusions in D correspond to arguments of our resulting SETAF (observe that each assumption $a \in \mathcal{A}$ is a conclusion of D); and
2. a set of conclusions C attacks a conclusion p in SF_D, i.e., $(C, p) \in R$, iff C contains a contrary for each set of assumptions S with $S \vdash p$, and C is $\subseteq$-minimal among all such sets (i.e., C is a *minimal hitting set* of the set $\{\{\overline{a} \mid a \in S\} \mid S \vdash p\}$).

Definition 3.17. For a given ABA instance $D = (\mathcal{L}, \mathcal{R}, \mathcal{A}, ^{-})$, let $\mathcal{S}_p = \{S \mid S \vdash p\}$ and $\overline{\mathcal{S}}_p = \{\{\overline{a} \mid a \in S\} \mid S \vdash p\}$ for each $p \in \mathcal{L}$. We construct the SETAF $SF^c_D = (A, R)$ with $A = \{p \mid \exists S \subseteq \mathcal{A} : S \vdash p\}$ and $R = \{(C, p) \mid p \in A, C \in HS_{min}(\overline{\mathcal{S}}_p)\}$.

Example 3.18. We construct SETAF SF^c_D from the ABA D from Example 3.4. The arguments in SF^c_D correspond to the conclusions in D, i.e., $A = \{a, b, c, p, q\}$. We determine the attackers of $p \in A$: first, we identify the set $\mathcal{S}_p = \{\{a\}, \{c\}\}$ that contains all assumption-sets that derive p (in D); the set $\overline{\mathcal{S}}_p = \{\{b\}, \{q\}\}$ contains the respective contraries. The unique hitting set of $\overline{\mathcal{S}}_p$ is $\{b, q\}$, thus $\{b, q\}$ attacks p. We depict the resulting SETAF below (the joint arcs from $\{b, q\}$ to p (in blue) represent the set-attack):

SETAF SF^c_D:



The construction indeed preserves the σ-conclusion-extensions for the considered semantics; moreover, we obtain the assumption-extensions of the original instance by projecting the conclusion-extensions to the assumptions $\mathcal{A}$.

Proposition 3.19. *For an ABA $D = (\mathcal{L}, \mathcal{R}, \mathcal{A}, ^{-})$, its associated SETAF SF^c_D and $\sigma \in \{grd, com, pref, stb\}$, it holds that $\sigma_{Th}(D) = \sigma(SF^c_D)$ and $\sigma(D) = \{C \cap \mathcal{A} \mid C \in \sigma(SF^c_D)\}$.*

3.3. Summary & Compatibility

We presented several different translations from ABA to CAFs and SETAFs and vice versa. For CAFs, we related claims with conclusions; for SETAFs, we considered two translations by relating arguments with assumptions and with conclusions, respectively.

When comparing the ABA instances when starting from a CAF or a SETAF (cf. Definition 3.9 and 3.13, respectively), we observe the following similarities: in both cases, the resulting ABA is flat, also, each rule contains only assumptions in its body, furthermore, no contrary of an assumption is an assumption. We furthermore observe the following notable difference between the two translations: while the translation from ABA to CAF potentially causes an exponential blow-up as the argument-construction can be exponential in the number of assumptions, we observe that the resulting SETAF is linear in the number of assumptions, i.e., the exponential blow-up can be avoided. We note, however, that the computation of the SETAF might be exponential—the computational effort is shifted to the construction of the attack relation which requires to identify tree-derivations $S \vdash \overline{a}$ in the ABA framework to define attacks (S, a) in the SETAF.

We end this section by presenting a strong intertranslatability result for our considered formalisms. For this, we make use of the translation from well-formed CAFs to SETAFs [8]. To fit our setting, we reformulate the translation in terms of hitting sets instead of CNF and DNF-formulas to capture the attack-structure of the frameworks.

Definition 3.20 (cf. [8])**.** For a well-formed CAF $\mathscr{F} = (A, R, cl)$, we define the corresponding SETAF $SF_{\mathscr{F}} = (A_{\mathscr{F}}, R_{\mathscr{F}})$ by letting $A_{\mathscr{F}} = cl(A)$ and $R_{\mathscr{F}} = \{(T, c) \mid c \in cl(A), T \in HS_{min}(\{cl(x_R^-) \mid x \in A, cl(x) = c\})\}$. For a SETAF $SF = (A, R)$, we define the corresponding CAF $\mathscr{F}_{SF} = (A_{SF}, R_{SF}, cl_{SF})$ with $A_{SF} = \{x_{c,h} \mid c \in A,\ h \in HS_{min}(c_R^-)\}$, $cl_{SF}(x_{c,h}) = c$, and $R_{SF} = \{(x_{c,h_x}, y_{d,h_y}) \mid c \in h_y\}$.

Restricting the translation to *redundancy-free* CAFs, i.e., frameworks s.t. there are no $x, y \in A$ with $cl(x) = cl(y)$, $x^+ = y^+$, and $x^- \subseteq y^-$, we obtain the following result.

Proposition 3.21. *Given an ABA $D = (\mathscr{L}, \mathscr{R}, \mathscr{A}, {}^-)$, its corresponding SETAF SF_D^c (cf. Definition 3.17), let $\mathscr{F}_D$ be the corresponding CAF (cf. Definition 3.6), and let $SF_{\mathscr{F}_D}^c$ be the SETAF corresponding to the CAF $\mathscr{F}_D$ (cf. Definition 3.20). It holds that $SF_D^c = SF_{\mathscr{F}_D}^c$.*

4. Strong Intertranslatability of LPs, CAFs, and SETAFs

In this section we strengthen the results regarding CAFs, LPs, and SETAFs by providing structure-preserving translations for suitable normal forms of the formalisms. This highlights their equivalent expressiveness. While there is an immediate correspondence between CAFs and LPs, the connection to SETAFs is via a detour making use of hitting sets, as we will explain in more detail in Section 4.1 (cf. [20]). The relations we will discuss are depicted in Figure 3. Our way to extract arguments from an LP is similar to the AF-instantiation reported in [15] where a semantics correspondence between LPs and AFs has been established. Due to space restrictions, we will focus our attention on stable semantics since this is the most commonly used semantics for LPs, but we want to emphasize that analogous results hold for the other cases, i.e. *com*, *grd*, and *pref* as well. Moreover, most results reported in this section are concerned with syntactical properties.

Logic Programs. We consider logic programs with default negation *not*. Such programs consist of rules of the form "$c \leftarrow a_1, \dots, a_n, \text{not } b_1, \dots, \text{not } b_m$." where $0 \leq n, m$ and the a_i, b_i and c are ordinary atoms. We let $head(r) = c$, $pos(r) = \{a_1, \dots, a_n\}$ and $neg(r) = \{b_1, \dots, b_m\}$. Let $\mathscr{L}(P)$ be the set of all atoms occurring in P. For $B = \{b_1, \dots, b_m\}$, we use not B as a shorthand for the conjunction not $b_1, \dots,$ not b_m. A rule r is *atomic* [6] if $pos(r) = \emptyset$; a program P is *atomic* if each rule in P is.

For LPs without default negation ($neg(r) = \emptyset$) the unique stable model is the smallest set of atoms closed under all rules, where a set E is closed under a rule r with $neg(r) = \emptyset$

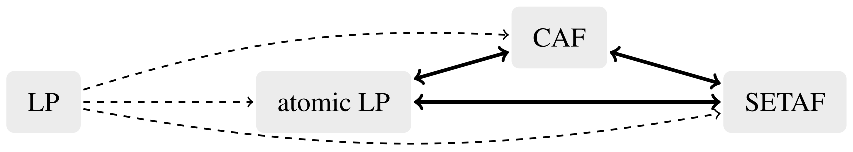


Figure 3. Transformations between formalisms discussed in Section 4

iff $pos(r) \subseteq E$ implies $head(r) \in E$. For any LP P, a set E of atoms is a *stable model* ($E \in stb(P)$) iff E is the stable model of $P^E = \{head(r) \leftarrow pos(r) \mid neg(r) \cap E = \emptyset\}$.

Example 4.1. If $P = \{(d \leftarrow \text{not}\, a,\ \text{not}\, b.), (d \leftarrow \text{not}\, c.), (a \leftarrow \text{not}\, c.), (c \leftarrow \text{not}\, a.), (b.)\}$, then P is atomic. For $E = \{b, c\}$ we have $P^E = \{(c.), (b.)\}$ and thus $E \in stb(P)$.

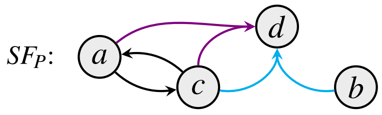
Redundancies. Throughout this section we will require redundancy notions for our formalisms. An argument $x \in A$ in a CAF $\mathscr{F} = (A, R, cl)$ is *redundant* if there is $y \in A$ with $cl(x) = cl(y)$ and $y^- \subseteq x^-$. An attack $(T, h) \in R$ in a SETAF $SF = (A, R)$ is *redundant* if there is $(T', h) \in R$ with $T' \subsetneq T$. A rule $r \in P$ of an atomic LP is *redundant* if there is $r' \in P$ with $head(r) = head(r')$ and $neg(r') \subseteq neg(r)$; an atom $a \in \mathscr{L}(P)$ is *redundant* if it does not occur as a rule head in P. A CAF resp. SETAF resp. LP without redundant arguments resp. attacks resp. rules and atoms is *redundancy-free*.

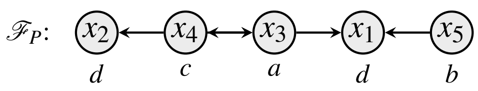
4.1. High Level Point of View

In the following subsections we will require various translations between the formalisms, which may appear rather technical at first glance. However, by closely inspecting all cases we observe that the constructed instances of the respective formalisms are quite similar in their spirit and translations are obtained by using suitably applied simple steps.

More precisely, inter-translating CAFs and atomic LPs is done by identifying rule heads with claims and bodies with in-going attacks. Recall our program P from above. In a rather immediate way, the program induces a CAF $\mathscr{F}_P$ consisting of five arguments x_i (one for each rule) where $cl(x_1) = cl(x_2) = d$, $cl(x_3) = a$, $cl(x_4) = c$, and $cl(x_5) = b$ corresponding to the rule heads. Moreover, $cl(x_1^-) = \{a, b\}$, $cl(x_2^-) = cl(x_3^-) = \{c\}$, $cl(x_4^-) = \{a\}$, and $cl(x_5^-) = \emptyset$ defines the attack relation of the well-formed CAF $\mathscr{F}_P$.

When connecting either CAFs or atomic LPs to SETAFs, the notion of a hitting set is required. In SETAFs, we do not use multiple copies of the same claim resp. rule head, but encode the acceptability condition solely in the attack relation. The corresponding SETAF would therefore possess only the four arguments a, b, c, d. For example, d *cannot* be accepted if (i) either a or b is inferred (first rule not applicable) and (ii) c is inferred (second rule not applicable either). This yields the following SETAF SF_P. Below, we also depict the CAF $\mathscr{F}_P$ we calculated earlier:





The more challenging part is dropping the assumption that the given LP P is atomic (see Figure 3). For this, we will utilize an inductive procedure constructing arguments [15].

Definition 4.2. For an LP P, A is an argument in P ($A \in Args(P)$) with $\textsc{Conc}(A) = c$, $\textsc{Rules}(A) = \bigcup_{i \leq n} \textsc{Rules}(A_i) \cup \{r\}$, and $\textsc{Vul}(x) = \bigcup_{i \leq n} \textsc{Vul}(A_i) \cup \{b_1, \ldots, b_m\}$ iff there are $A_1, \ldots, A_n \in Args(P)$ and a rule $r \in P$ with $r = c \leftarrow \textsc{Conc}(A_1), \ldots, \textsc{Conc}(A_n)$, not $b_1, \ldots,$ not b_m, and $r \notin \textsc{Rules}(A_i)$ for all $i \leq n$.

We will show that this procedure can be mimicked by rewriting P. For example let $P' = \{(d \leftarrow c,\ \text{not}\, b.), (d \leftarrow \text{not}\, c.), (a \leftarrow \text{not}\, c.), (c \leftarrow \text{not}\, a.), (b.)\}$. The atomic program P from above is the result of inserting the rule $(c \leftarrow \text{not}\, a.)$ in $(d \leftarrow c,\ \text{not}\, b.)$.

4.2. Translations

CAFs and Logic Programs. We will now formally establish the correspondence between CAFs and LPs, by making use of $Args(P)$ in case P is not atomic.

Definition 4.3. For a CAF $\mathscr{F} = (A, R, cl)$, we define the corresponding atomic LP $P_{\mathscr{F}}$ by $P = \{c \leftarrow \text{not}\, B. \mid a \in A,\ cl(a) = c,\ cl(a^-) = B\}$. For an LP P, we set $\mathscr{F}_P = (A_P, R_P, cl_P)$ where $A_P = Args(P)$, $R_P = \{(a,b) \mid cl(a) \in \textsc{Vul}(b)\}$, and $cl_P(a) = \textsc{Conc}(a)$.

Example 4.4. The LP P' from above yields four arguments stemming from the atomic rules, e.g. there is some argument A with $\textsc{Conc}(A) = c$, $\textsc{Vul}(A) = \{a\}$ and $\textsc{Rules}(A) = \{(c \leftarrow \text{not}\, a.)\}$. From $(d \leftarrow c,\ \text{not}\, b.)$ and this argument A we construct another argument with conclusion d and vulnerabilities $\{a, b\}$ (inherited from A and the applied rule). The complete corresponding CAF $\mathscr{F}_{P'}$ is the same as the CAF $\mathscr{F}_P$ depicted in Section 4.1.

A rather convenient feature of this approach is that we can infer the semantics correspondence from [15] due to the way CAF semantics make use of the claims.

Proposition 4.5. *For $\mathscr{F}$ a CAF and P an LP, $stb(\mathscr{F}) = stb(P_{\mathscr{F}})$ and $stb(P) = stb(\mathscr{F}_P)$.*

By inspecting Definition 4.2 we observe that the challenging part is handling positive atoms in rule bodies. If P is atomic, we can extract the corresponding CAF $\mathscr{F}_P = (A_P, R_P, cl_P)$ immediately via $A_P = P$, $R_P = \{(a,b) \mid head(a) \in neg(b)\}$, and $cl_P(a) = head(a)$. The fact that atomic LPs and CAFs are so closely related motivates the question whether we can transform the LP before constructing the arguments as done in [15]. A technique of this kind could pre-process the LP instead of utilizing the instantiation procedure. In the following, we formalize this idea.

Definition 4.6. For an LP P the corresponding atomic LP P_{AT} is defined inductively:

- If $r \in P$ is atomic, then $r \in P_{AT}$.
- If there is a rule $r_0 \in P$ with $pos(r_0) = \{a_1, \dots, a_n\}$ and for each a_i, $1 \le i \le n$, there is some rule $r_i \in P_{AT}$ s.t. $head(r_i) = a_i$, then there is a rule $r \in P_{AT}$ with $head(r) = head(r_0)$, $pos(r) = \emptyset$, and $neg(r) = \bigcup_{i=1}^{n} neg(r_i)$.

Example 4.7. Applied to our LP $P' = \{(d \leftarrow c,\ \text{not}\, b.), (d \leftarrow \text{not}\, c.), (a \leftarrow \text{not}\, c.), (c \leftarrow \text{not}\, a.), (b.)\}$ this procedure yields $P'_{AT} = P$ with P as in Example 4.1.

The following theorem formalizes that this pre-processing step successfully mimics the inductive procedure from [15]. Informally speaking, instantiating the LP is done as in Definition 4.2 and yields the same result as turning the LP into an atomic one via iterative insertion of atomic rules and then extracting the corresponding CAF by identifying rule heads with claims and rule bodies with in-going attacks. Formally:

Theorem 4.8. *Let P be an LP. Then $\mathscr{F}_P = \mathscr{F}_{P_{AT}}$.*

SETAFs and LPs We also want to briefly mention that analogous results hold when turning an LP into a SETAF, which can be done as follows. For an LP P we define by $A_P = \bigcup_{A \in Args(P)} \textsc{Conc}(a)$ and $R_P = \{(T, c) \mid T \in HS_{min}(\{\textsc{Vul}(A) \mid A \in Args(P),\ \textsc{Conc}(A) = c\})\}$ the associated SETAF SF_P. For a SETAF $SF = (A, R)$, we define its associated LP $P_{SF} = \{c \leftarrow \text{not}\, B. \mid B \in HS_{min}(c_R^-)\}$. As observed before, the construction of $Args(P)$ can be omitted if P is atomic. With these constructions, we find:

Theorem 4.9. *For a SETAF SF and an LP P, $stb(SF) = stb(P_{SF})$ and $stb(P) = stb(SF_P)$. Moreover, $SF_P = SF_{P_{AT}}$.*

4.3. Summary & Compatibility

In this section, we presented translations from LPs to SETAFs and to CAFs, respectively. We observe that when instantiating an LP as CAF or SETAF, an exponential blow-up cannot be avoided due to the construction of arguments which is an inherent part of both procedures. For atomic LPs, on the other hand, the number of arguments is linear in the number of rules in both formalisms. For the other direction, i.e., when translating a CAF or SETAF into an LP, the resulting LP is atomic. It can be shown that for atomic LPs, these constructions are bijective and each others inverse, establishing a close relation.

Lemma 4.10. *For all redundancy-free atomic LPs P, CAFs $\mathcal{F}$, and SETAFs SF, respectively, it holds that i) $SF_{P_{SF}} = SF$; ii) $P_{SF_P} = P = P_{\mathcal{F}_P}$; and iii) $\mathcal{F}_{P_{\mathcal{F}}} = \mathcal{F}$.*

We end this section with a strong intertranslatability result in the spirit of Theorem 3.21, stating that all (atomic, well-formed, and redundancy-free) instances of the considered formalisms can be equivalently represented as CAFs, LPs, or SETAFs without any loss of information via the presented translations and the method in [9] (cf. Definition 3.20). This shows that all of our constructions are compatible with each other and similar in their behavior. In particular, the order in which they are applied is arbitrary.

Theorem 4.11. *For all redundancy-free atomic LPs P, SETAFs SF, CAFs $\mathcal{F}$, we have $\mathcal{F}_{SF} \cong \mathcal{F}_{P_{SF}}$; $P_{SF} = P_{\mathcal{F}_{SF}}$; $\mathcal{F}_P \cong \mathcal{F}_{SF_P}$; $SF_P = SF_{\mathcal{F}_P}$; $SF_{\mathcal{F}} = SF_{P_{\mathcal{F}}}$; and $P_{\mathcal{F}} = P_{SF_{\mathcal{F}}}$.*

5. Discussion

In this paper we investigated translations between the argumentation formalisms ABA, CAF, SETAF as well as their connections to LP. We strengthened the implicitly existing intertranslatability result by providing additional translations, filling some of the existing gaps. For selected translations we showed structure-preserving properties and argued why others (such as those involving ABA) might not feature this preservation. Finally, our overview yields implications regarding expressiveness: the formalisms under our consideration admitting strong intertranslatability are equally expressive—i.e. , they can describe the same sets of models (extensions). These results illustrate the usefulness of the versatility in argumentation formalisms: while certain applications might suggest the usage of a specific formalism, it might be useful to later translate this framework and utilize features that are native to another formalism. Strong intertranslatability even guarantees the preservation of the structure, which opens interesting topics for future work: as some of the discussed translations are modular in some sense, one might even be able to instantiate the same knowledge base as part formalism A and part formalism B, while connecting both parts in later steps during the workflow. Another useful consequence of our findings is that it is now easier to transfer concepts and ideas between formalisms, serving as a starting point for various investigations that highlight the similarities of the considered approaches even further.

Acknowledgments

This research has been supported by the Vienna Science and Technology Fund (WWTF) through project ICT19-065, the Austrian Science Fund (FWF) through projects W1255-N23 and P32830, and by the German Federal Ministry of Education and Research (BMBF, 01/S18026A-F) by funding the competence center for Big Data and AI "ScaDS.AI" Dresden/Leipzig.

References

[1] Dung PM. On the Acceptability of Arguments and its Fundamental Role in Nonmonotonic Reasoning, Logic Programming and n-Person Games. Artif Intell. 1995;77(2):321-58.
[2] Brewka G, Woltran S. Abstract Dialectical Frameworks. In: Proceedings of KR 2010. AAAI Press; 2010. p. 780-5.
[3] Bondarenko A, Dung PM, Kowalski RA, Toni F. An Abstract, Argumentation-Theoretic Approach to Default Reasoning. Artif Intell. 1997;93:63-101.
[4] Dvořák W, Woltran S. Complexity of abstract argumentation under a claim-centric view. Artif Intell. 2020;285:103290.
[5] Nielsen SH, Parsons S. A Generalization of Dung's Abstract Framework for Argumentation: Arguing with Sets of Attacking Arguments. In: Proceedings of ArgMAS 2006. Springer; 2006. p. 54-73.
[6] Janhunen T. Representing Normal Programs with Clauses. In: Proceedings of ECAI 2004. IOS Press; 2004. p. 358-62.
[7] Caminada M, Schulz C. On the Equivalence between Assumption-Based Argumentation and Logic Programming. J Artif Intell Res. 2017;60:779-825.
[8] Dvořák W, Rapberger A, Woltran S. Argumentation Semantics under a Claim-centric View: Properties, Expressiveness and Relation to SETAFs. In: Proceedings of KR 2020; 2020. p. 341-50.
[9] Dvořák W, Rapberger A, Woltran S. On the Relation Between Claim-Augmented Argumentation Frameworks and Collective Attacks. In: Proceedings of ECAI 2020. vol. 325. IOS Press; 2020. p. 721-8.
[10] Dvořák W, Keshavarzi Zafarghandi A, Woltran S. Expressiveness of SETAFs and Support-Free ADFs Under 3-Valued Semantics. In: Proceedings of COMMA 2020. IOS Press; 2020. p. 191-202.
[11] Alcântara JFL, Sá S. Equivalence Results between SETAF and Attacking Abstract Dialectical Frameworks. In: Proceedings NMR 2021; 2021. p. 139-48.
[12] Polberg S. Developing the Abstract Dialectical Framework [PhD Thesis]. Vienna University of Technology, Institute of Information Systems; 2017.
[13] Strass H. Approximating operators and semantics for abstract dialectical frameworks. Artif Intell. 2013;205:39-70.
[14] Alcântara JFL, Sá S, Guadarrama JCA. On the Equivalence Between Abstract Dialectical Frameworks and Logic Programs. Theory Pract Log Program. 2019;19(5-6):941-56.
[15] Caminada M, Sá S, Alcântara J, Dvořák W. On the equivalence between logic programming semantics and argumentation semantics. Int J Approx Reasoning. 2015;58:87-111.
[16] Berge C. Hypergraphs - combinatorics of finite sets. vol. 45 of North-Holland mathematical library. North-Holland; 1989.
[17] Cyras K, Fan X, Schulz C, Toni F. Assumption-Based Argumentation: Disputes, Explanations, Preferences. In: Handbook of Formal Argumentation. College Publications; 2018. p. 365-408.
[18] Bikakis A, Cohen A, Dvořák W, Flouris G, Parsons S. Joint Attacks and Accrual in Argumentation Frameworks. FLAP. 2021;8(6):1437-501.
[19] Flouris G, Bikakis A. A comprehensive study of argumentation frameworks with sets of attacking arguments. Int J Approx Reason. 2019;109:55-86.
[20] Dvořák W, König M, Ulbricht M, Woltran S. A Reduct-Driven Study of Argumentation Frameworks With Collective Attacks. In: Proceedings of NMR 2021; 2021. p. 285-94.