On Independence and SCC-Recursiveness in Assumption-Based Argumentation

Lydia Blümel[1] , **Anna Rapberger**[2] , **Matthias Thimm**[1] and **Francesca Toni**[2]
[1]Artificial Intelligence Group, University of Hagen
[2]Imperial College London
{lydia.bluemel, matthias.thimm}@fernuni-hagen.de, {a.rapberger,ft}@imperial.ac.uk

Abstract

We introduce a notion of conditional independence in (flat) assumption-based argumentation (ABA), where independence between (sets of) assumptions amounts to the presence of information about one set of assumptions not impacting the acceptability of another. We study general properties, computational complexity, and the relation to independence in abstract argumentation. In light of the high computational complexity of deciding independence, we introduce sound methods for checking independence in polynomial time via two different routes: the first utilizes the strongly connected components (SCCs) of the instantiated abstract argumentation framework; the second exploits the structure of the ABA framework directly. Along the way, we introduce the notion of SCC-recursiveness for ABA.

1 Introduction

The ability to recognize and manage (in)dependence is crucial in symbolic reasoning at large [Darwiche and Pearl, 1994; Darwiche, 1997; Lang *et al.*, 2002; Rienstra *et al.*, 2020]. It plays an essential role in causal reasoning [Pearl, 2009], e. g., in probabilistic models, where (in)dependence between variables is built into graphical representations such as Bayesian networks [Geiger *et al.*, 1990]. The ability to determine independence contributes to the explainability [Halpern and Pearl, 2001] and trustworthiness [Belle *et al.*, 2024] of reasoning systems, while also improving efficiency by empowering breaking problems down so that they can be managed more effectively.

In this paper, we study conditional independence in computational argumentation, which captures how different parts of the argumentative information can be logically separated. Two prominent areas therein are abstract argumentation, coined by the seminal paper of Dung (1995), and structured argumentation, as overviewed by Besnard *et al.* (2014b). Unlike abstract argumentation, where arguments are considered abstract, structured argumentation frameworks focus on the structure of the arguments, enabling a finer-grained analysis of the arguments and relations. *Assumption-based argumentation (ABA)* [Bondarenko *et al.*, 1997; Čyras *et al.*, 2018] is a well-known form of structured argumentation, whose building blocks are assumptions (defeasible elements) and inference rules. We illustrate this with the following example.

Example 1.1. *Alice, Bob, and Carol plan a tandem trip; we consider assumptions a (*Alice cycles*), b (*Bob cycles*), and c (*Carol cycles*) for each of our protagonists. Naturally, only two of them cycle at the same time, that is, not all our assumptions can be true at once. We capture the relations between our assumptions with inference rules; e. g., "if Bob and Carol cycle then Alice does not" is captured by the rule ($\overline{a} \leftarrow b, c$), here, $\overline{a}$ is the contrary of a. Crucially, if the* weather is bad *(d) nobody can cycle. On the bright side, then there is no need to* water the plants *(w) outside. Also, Alice thinks about* bringing her book *(k), but if she cycles, it might be too heavy.*[1]

ABA has been broadly studied computationally [Toni, 2013; Cyras *et al.*, 2021a; Craven and Toni, 2016; Lehtonen *et al.*, 2024a; Lehtonen *et al.*, 2024b]. ABA is also widely applicable, e.g., in healthcare [Cyras *et al.*, 2021b], to provide explanations [Cyras *et al.*, 2021a], for causal discovery [Russo *et al.*, 2024], and planning [Fan, 2018]. In all these settings, a good understanding of independence between various components in ABA frameworks (ABAFs) is crucial. However, to date the study of independence within ABA remains underexplored. Actually, with the notable exception of the conditional independence analysis in abstract argumentation [Rienstra *et al.*, 2020], the topic has been largely neglected in the computational argumentation literature so far.

Turning our attention to ABA, we address the following question: *under what conditions are two (sets of) assumptions independent from each other, relative to a (potentially empty) set of assumptions that is considered prior knowledge?* In other words, does the choice of truth values for one set of assumptions restrict the available choices for the truth values of another set, given the truth values for a third set are fixed?

Example 1.2 (Example 1.1 cont.)**.** *Let us find out if Alice and Bob influence each other's cycling activities. First, are a and b (unconditionally) independent of each other? The answer to that question is positive: knowing that Alice cycles does not give us any information about Bob, Alice could cycle with either Bob or Carol; similarly, if Alice does not cycle we do not know whether Bob cycles; either Bob and Carol*

[1]A formalization is given in Example 3.2.

cycle together or the weather could have ruined their trip. In contrast, a and b depend on each other, given c: knowing that Carol cycles implies that one of the other two cycles as well.

Note that w and k depend on each other, because there is no way to reject both of them at the same time. They are independent when given information about $\{a\}$, $\{d\}$, or $\{b, c\}$.

Identifying conditional independence in ABA is computationally challenging, as we will show. To alleviate the high computational complexity we exploit the structure of ABA and explore *SCC-recursiveness* for ABAFs. Rienstra et al. (2020) demonstrate the advantages of SCC-recursiveness for efficiently checking conditional independence between arguments, based on the SCCs of an abstract argumentation framework (AF) [Dung, 1995]. Despite these considerable computational advantages, SCC-recursiveness has not received attention in the context of structured argumentation so far. In the scope of our independence studies for ABA, we tackle this issue and propose an SCC-recursive schema for ABA semantics alongside sound independence checks for assumptions in ABAFs.

Overall, we make the following contributions, for a well-studied fragment of ABA whereby assumptions cannot be inferred (known as *flat* ABA) and where assumptions, their contraries and any sentences in inference rules are atomic.

- We introduce conditional independence in ABA and study its properties. We analyze its computational complexity and reveal the close relation of independence in ABAFs and AFs. We also settle the complexity of the corresponding decision problems for AFs. Section 4
- We introduce sound and poly-time methods for checking independence in ABA via two different routes:
  (1) applying the approach by Rienstra et al. (2020) to a suitable AF instantiation for ABAFs; Section 5.1
  (2) exploiting the structure of the ABAF directly, based on the SCCs of the so-called dependency graph [Rapberger *et al.*, 2022]. Section 5.3
- To facilitate route (2), we introduce SCC-recursiveness for ABA and show that all semantics under consideration satisfy this property. Section 5.2

Due to space restrictions, we focus on complete-based semantics; analogous results for admissible semantics, alongside with proofs and further discussions are included in an extended version found online (DOI 10.5281/zenodo.15470789, https://zenodo.org/records/15470789).

2 Related Work

To address the challenges of computationally demanding dependency models, researchers [Verma and Pearl, 1988; Rienstra *et al.*, 2020] have proposed sound (but potentially incomplete) Directed Acyclic Graph (DAG) representations to check independence. In the context of computational argumentation, Rienstra et al. (2020) identify a way to transform AFs into DAGs, based on their SCCs, to express (some) dependencies between arguments. Our study of the relation between SCCs in ABA and independence is inspired by this work. However, differently from their work, we focus on structured argumentation in the form of ABA.

Heyninck (2023) proposes a generic framework for studying independence in approximation fixpoint theory, applicable to any logic with fixpoint semantics, including logic programs under partial stable models and well-founded model semantics. Since the latter are instances of ABA in general, and of the restricted form of ABA we consider, the resulting framework in [Heyninck, 2023] is applicable to our setting, too. That being said, our focus lies on the relation to abstract argumentation, and the relation between independence and SCCs (which we newly introduce in this paper for ABA).

Some works use reasoning with independence information to perform causal discovery with ABA [Russo *et al.*, 2024] or paradigms related to ABA, such as ASP [Zhalama *et al.*, 2019]. This line of work is orthogonal to ours, as we focus on studying independence in ABA, rather than using ABA for ascertaining independence for causal discovery. Another work [Besnard *et al.*, 2014a] defines an approach to reasoning about causes via argumentation, which, like in our case, is structured but built on classical logic and including ontologies. It does not consider independence in argumentation.

3 Background

A directed graph is a pair $G = (V, E)$ where V is a set of vertices and $E \subseteq V^2$. A (un)directed path p is a sequence $v_1 \dots v_n$, $v_i \in V$, $v_i \neq v_j$, $i \neq j$, with $(v_i, v_{i+1}) \in E$ (or $(v_{i+1}, v_i) \in E$); a cycle is a sequence $v_n v_1 \dots v_n$ where $v_1 \dots v_n$ is a directed path. G is a directed acyclic graph (DAG) if it contains no directed cycles. For a vertex set $U \subseteq V$, $PA_G(U) = \{w \in V \mid (w, u) \in E, u \in U\} \setminus U$ are the parents of U, $AN_G(U) = \{w \in V \mid \exists u \in U, p = (w, ..., u) \text{ directed path}\} \setminus U$ the ancestors of U. We say $v \in V$ is a descendant of U, if $U \cap AN_G(\{v\}) \neq \emptyset$. We denote by $DD_G(v)$ the set of all descendants and by $ND_G(v)$ the set of all non-descendants and non-parents of U.

3.1 Assumption-based Argumentation

We recall assumption-based argumentation (ABA) [Čyras *et al.*, 2018]. We assume a deductive system $(\mathcal{L}, \mathcal{R})$, where $\mathcal{L}$ is a formal language, i.e., a set of sentences, and $\mathcal{R}$ is a set of inference rules over $\mathcal{L}$. A rule $r \in \mathcal{R}$ has the form $a_0 \leftarrow a_1, \dots, a_n$ with $a_i \in \mathcal{L}$. We write $head(r) = a_0$ and $body(r) = \{a_1, \dots, a_n\}$ for the possibly empty body of r.

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

Unless otherwise specified, we assume a unique contrary a_c for each assumption $a \in \mathcal{A}$. We write $\overline{a}$ to denote a_c.

Example 3.2. *We formalise the introductory Example 1.1 as ABAF D with assumptions $\{a, b, c, d, w, k\}$, their contraries, and inference rules, $(\overline{v} \leftarrow d)$, $(\overline{d} \leftarrow v)$ for $v \in \{a, b, c\}$, and*

$$\overline{a} \leftarrow b, c \qquad \overline{b} \leftarrow a, c \qquad \overline{c} \leftarrow a, c \qquad \overline{k} \leftarrow a \qquad \overline{w} \leftarrow d$$

Below, we fix an arbitrary ABAF $D = (\mathcal{L}, \mathcal{R}, \mathcal{A}, ^-)$. A sentence $p \in \mathcal{L}$ is *tree-derivable* from assumptions $A \subseteq \mathcal{A}$ and rules $R \subseteq \mathcal{R}$, denoted by $A \vdash_R p$ (tree-derivation), if there is a finite rooted labelled tree T such that the root is labelled with p, the set of labels for the leaves of T is equal to

A or $A \cup \{\top\}$, and there is a surjective mapping from the set of internal nodes to R, satisfying that for each internal node v, there is $r \in R$, s.t. v is labelled with $head(r)$, and the set of all successor nodes corresponds to $body(r)$ or $\top$ if $body(r) = \emptyset$. We say p is derivable (in D) iff there is a tree-derivation $A \vdash_R p$, $A \subseteq \mathcal{A}$, $R \subseteq \mathcal{R}$. We call $A \vdash p$ an *(ABA) argument* if there is $R \subseteq \mathcal{R}$ s.t. $A \vdash_R p$. Arguments of the form $\{a\} \vdash a$, $a \in \mathcal{A}$ are assumption arguments, arguments with $R \neq \emptyset$ rule-based arguments. Each assumption argument is unique; we identify assumption arguments with their assumptions.

By $Th_D(A) = \{p \mid \exists A \vdash p\}$ we denote the set of all claims derivable from assumptions A in D. We let $\overline{A} = \{\overline{a} \mid a \in A\}$. The set A *attacks* $B \subseteq \mathcal{A}$ if there is $A' \subseteq A$, $b \in B$ s.t. $A' \vdash \overline{b}$. A is conflict-free if it does not attack itself; A is admissible if it is conflict-free and defends itself.

A semantics σ is a function assigning ABAF a set of σ-labellings. We recall complete, grounded, preferred, and stable semantics (abbr. co, gr, pr, st) [Schulz and Toni, 2017].

Definition 3.3. *A labelling is a (partial) function $\lambda : \mathcal{A} \to L$, $L = \{\texttt{in}, \texttt{out}, \texttt{und}\}$. For $A \subseteq \mathcal{A}$, we let $\lambda_A : A \to L$ denote a partial labelling of A; $\lambda|_A$ denotes the restriction of λ to A. For $lab \in L$, $lab(\lambda) = \{x \in \mathcal{A} \mid \lambda(x) = lab\}$.*

Definition 3.4. *$\lambda : \mathcal{A} \to L$ is a* complete *labelling (abbr. co-labelling) of an ABAF $D = (\mathcal{L}, \mathcal{A}, \mathcal{R}, \overline{})$ iff for each $a \in \mathcal{A}$,*

- *$\lambda(a) = \texttt{in}$ iff for all $B \vdash \overline{a}$, there is $b \in B$ s.t. $\lambda(b) = \texttt{out}$;*
- *$\lambda(a) = \texttt{out}$ iff there is $B \vdash \overline{a}$ s.t. $\lambda(b) = \texttt{in}$ for all $b \in B$.*

A co-labelling λ is grounded *iff $\texttt{und}(\lambda)$ is $\subseteq$-maximal among all co-labellings of F;* preferred *iff $\texttt{in}(\lambda)$ is $\subseteq$-maximal among all co-labellings of F; and* stable *iff $\texttt{und}(\lambda) = \emptyset$.*

For semantics σ, the set Λ_σ denotes the set of all σ-labellings of ABAF D. For $A \subseteq \mathcal{A}$, we say λ_A is σ-compatible with D if there is $\lambda \in \Lambda_\sigma(D)$ s.t. $\lambda|_A = \lambda_A$.

Assumption 3.5. *We focus on ABAFs which are* flat*, i. e., for each rule $r \in \mathcal{R}$, $head(r) \notin \mathcal{A}$ (no assumption can be derived, and* finite*, i. e., $\mathcal{L}$, $\mathcal{R}$, $\mathcal{A}$ are finite; moreover, each sentence in $s \in \mathcal{L}$ is atomic and derivable, i. e., there is $A \subseteq \mathcal{A}$ s.t. $A \vdash s$, and each rule is stated explicitly (given as input).*

ABAFs and Abstract Argumentation An argumentation framework (AF) [Dung, 1995] is a directed graph $F = (\mathbb{A}, \mathbb{R})$ where $\mathbb{A} \subseteq \mathcal{U}$ are arguments and $\mathbb{R} \subseteq \mathbb{A}^2$ are *attacks*. We fix a non-finite set $\mathcal{U}$ of arguments. For $x, y \in A$, if $(x, y) \in R$ we say x *attacks* y; $E \subseteq \mathbb{A}$ *defends* x if it attacks each attacker of x; E is *conflict-free* if it does not attack itself; and *admissible* if it is conflict-free and defends all $x \in E$.

We recall labelling-based complete, grounded, preferred, and stable semantics (abbr. co, gr, pr, st) for AFs, following [Caminada and Pigozzi, 2011]. Analogous to ABAFs, a labelling is a function $\lambda : \mathbb{A} \to L$ (cf. Definition 3.3).

Definition 3.6. *A labelling $\lambda : \mathbb{A} \to L$ is a* complete *labelling (abbr. co-labelling) of an AF $F = (\mathbb{A}, \mathbb{R})$ iff for each $x \in \mathbb{A}$,*

- *$\lambda(x) = \texttt{in}$ iff $\lambda(y) = \texttt{out}$ for every attacker y of x;*
- *$\lambda(x) = \texttt{out}$ iff $\lambda(y) = \texttt{in}$ for some attacker y of x.*

A co-labelling λ is grounded *iff $\texttt{und}(\lambda)$ is $\subseteq$-maximal among all co-labellings of F;* preferred *iff $\texttt{in}(\lambda)$ is $\subseteq$-maximal among all co-labellings of F; and* stable *iff $\texttt{und}(\lambda) = \emptyset$.*

We use Λ_σ, analogous to ABAFs (cf. below Definition 3.4).

ABAFs are closely related to AFs [Čyras *et al.*, 2018].

Definition 3.7. *The associated AF $F_D = (\mathbb{A}, \mathbb{R})$ of an ABAF $D = (\mathcal{L}, \mathcal{R}, \mathcal{A}, \overline{})$ is given by $\mathbb{A} = \{A \vdash p \mid \exists R \subseteq \mathcal{R} : A \vdash_R p\}$ and attack relation $(A \vdash p, A' \vdash p') \in R$ iff $p \in \overline{A'}$.*

The translation preserves the semantics [Schulz and Toni, 2017], the correspondence is 1:1 for all considered semantics.

Proposition 3.8. *Let D be an ABAF with AF F_D, $\sigma \in \{gr, co, pr, st\}$. If $\lambda \in \Lambda_\sigma(D)$ then $(\{A \vdash p \mid A \in \texttt{in}(\lambda)\}, \{A \vdash p \mid \exists a \in A : a \in \texttt{out}(\lambda)\}, \{A \vdash p \mid \exists a \in A : a \in \texttt{und}(\lambda), A \cap \texttt{out}(\lambda) = \emptyset\}) \in \Lambda_\sigma(F)$; if $\lambda \in \Lambda_\sigma(F)$ then $(\texttt{in}(\lambda|_{\mathcal{A}}), \texttt{out}(\lambda|_{\mathcal{A}}), \texttt{und}(\lambda|_{\mathcal{A}})) \in \Lambda_\sigma(D)$.*

Each AF induces an ABAF by associating assumptions and arguments [Toni, 2012]. The translation preserves semantics.

Definition 3.9. *For an AF $F = (\mathbb{A}, \mathbb{R})$, we define the ABAF $D_F = (\mathcal{L}, \mathcal{A}, \mathcal{R}, \overline{})$ s.t. $\mathcal{A} = \mathbb{A}$, and $\overline{a} \leftarrow b \in \mathcal{R}$ iff $(b, a) \in \mathbb{R}$.*

Proposition 3.10. *Let F be an AF, D_F its associated ABAF, and $\sigma \in \{co, pr, gr, st\}$. Then, $\Lambda_\sigma(F) = \Lambda_\sigma(D_F)$.*

3.2 Conditional Independence

A *dependency model* I over a set of variables V is a ternary relation over disjoint subsets of V. We write $X \perp_I Y \mid Z$ iff $(X, Y, Z) \in I$ and say X *and* Y *are independent, given* Z [Pearl and Paz, 1986]. We drop I if clear from context.

We recall semi-graphoid axioms [Pearl, 2009]:

Symmetry $A \perp B \mid C \Rightarrow B \perp A \mid C$

Decomposition $A \perp B \cup B' \mid C \Rightarrow A \perp B \mid C$

Weak Union $A \perp B \cup B' \mid C \Rightarrow A \perp B \mid C \cup B'$

Contraction $A \perp B \mid C \wedge A \perp B' \mid C \cup B \Rightarrow A \perp B \cup B' \mid C$

For a DAG, the notion of *d-separation* introduces a dependency model which is a semi-graphoid; also, checking d-separation is in P [Darwiche, 2009]. A node $v = v_i$ in path $v_1 \ldots v_n$ is a collider iff $(v_{i-1}, v_i), (v_{i+1}, v_i) \in E$.

Definition 3.11. *Let $G = (V, E)$ be a DAG and $A, B, C \subseteq V$. Then, A and B are* d-separated*, given C ($A \perp_d B \mid C$) in G, iff for all undirected paths p between A and B, it holds that p contains a collider v s.t. $(\{v\} \cup DD_G(v)) \cap C = \emptyset$ or $C \cap \{v \in p \mid v$ is not a collider$\} \neq \emptyset$.*

Rienstra *et al.* [2020] introduce independence for AFs.

Definition 3.12. *For an AF $F = \mathbb{A}, \mathbb{R}$, semantics σ and disjoint sets $A, B, C \subseteq \mathbb{A}$ we say A is* σ-independent of B, given C in F, *written $A \perp_\sigma B \mid C$, iff, for all $\lambda_1, \lambda_2 \in \Lambda_\sigma(F)$, if $\lambda_1|_C = \lambda_2|_C$ then there is some $\lambda_3 \in \Lambda_\sigma(F)$ s.t. $\lambda_3|_A = \lambda_1|_A$, $\lambda_3|_B = \lambda_2|_B$ and $\lambda_3|_C = \lambda_1|_C = \lambda_2|_C$.*

4 Independence in ABA

We investigate a semantical notion of conditional independence between assumptions. Similar to [Rienstra *et al.*, 2020; Darwiche, 1997], we base our notion on the acceptance status of assumptions. Intuitively, two sets of assumptions A and B are independent of each other, given a third set C, if coming to know the acceptance status of A in addition to knowing the acceptance status of C does not influence the acceptance status of B.

Definition 4.1. *Let $D = (\mathcal{L}, \mathcal{R}, \mathcal{A}, ^-)$ be an ABAF, σ be a semantics, and let $A, B, C \subseteq \mathcal{A}$ be disjoint sets of assumptions. Then A* is σ-independent of B, given C *in D, written $A \perp_\sigma B \mid C$, iff, for all labellings $\lambda_1, \lambda_2 \in \Lambda_\sigma(D)$, if $\lambda_1|_C = \lambda_2|_C$ then there is some labelling $\lambda_3 \in \Lambda_\sigma(D)$ s.t. $\lambda_3|_A = \lambda_1|_A$, $\lambda_3|_B = \lambda_2|_B$ and $\lambda_3|_C = \lambda_1|_C = \lambda_2|_C$.*

We drop σ and write $A \perp B \mid C$, if clear from context. By I_σ we denote the corresponding dependency model.

Example 4.2. *The ABAF D from Example 3.2 has five co-labellings; four of them are preferred and stable:*

σ-lab	a	b	c	d	w	k
co, gr	und	und	und	und	und	und
co, pr, st	in	in	out	out	in	out
co, pr, st	in	out	in	out	in	out
co, pr, st	out	in	in	out	in	in
co, pr, st	out	out	out	in	out	in

We can infer several (in)dependencies here.

- *The assumptions a and b are σ-independent wrt. $\emptyset$ for $\sigma \in \{pr, st\}$. For this, we identify the labels which are individually assigned to a and b under σ and verify that each combination of them is realized under σ.*
- *a and b are not co-independent, though: assigning a the label* `in` *and b the label* `und` *is individually possible, but there is no labelling with $\lambda(a) = \texttt{in}$ and $\lambda(b) = \texttt{und}$.*
- *When conditioning on $\{d\}$, the assumptions a and b are dependent under pr and st semantics. If $\lambda(d) = \texttt{out}$, then a and b can be individually* `out`*, but not together.*

Our first central result sheds light on the close connection between assumption independence and independence between arguments (cf. [Rienstra *et al.*, 2020]). We show that two sets of assumptions A, B are independent, given C, iff their corresponding assumption arguments are. Below, we identify assumption argument $\{a\} \vdash a$ with assumption a.

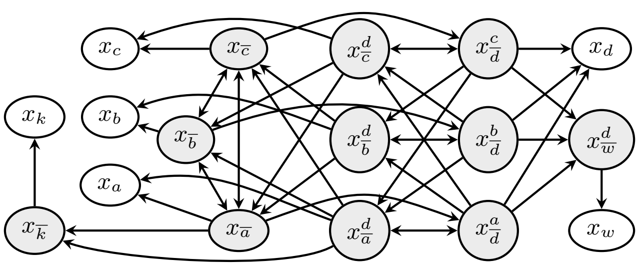
Proposition 4.3. *Let $D = (\mathcal{L}, \mathcal{R}, \mathcal{A}, ^-)$ be an ABAF, F_D its corresponding AF, and let $\sigma \in \{co, gr, pr, st\}$. Then, for all $A, B, C \subseteq \mathcal{A}$, we have $A \perp_\sigma B \mid C$ in D iff $A \perp_\sigma B \mid C$ in F_D.*

Example 4.4. *Consider the ABAF from Example 4.2. We obtain an AF F_D with assumption arguments $x_v : \{v\} \vdash v$ for $v \in \{a, b, c, d, w, k\}$ and rule-based arguments*

$$x_{\overline{k}} : \{a\} \vdash \overline{k} \quad x_{\overline{a}} : \{b, c\} \vdash \overline{a} \quad x_{\overline{b}} : \{a, c\} \vdash \overline{b} \quad x_{\overline{c}} : \{a, c\} \vdash \overline{c}$$

$$x_{\overline{w}} : \{d\} \vdash \overline{w} \quad \forall v \in \{a, b, c\} : \quad x^d_{\overline{v}} : \{d\} \vdash \overline{v} \quad x^v_{\overline{d}} : \{v\} \vdash \overline{d}$$

We have $(x, y) \in R$ iff $x = A \vdash p, y = B \vdash q$, and for some $b \in B : \overline{b} = p$. We depict the corresponding AF F_D below.



By Proposition 4.3 we can use the AF labellings to check for independence between assumptions; e.g., since x_k and x_a are pr-dependent in F_D, k and a are pr-dependent in D.

As a consequence, we can transfer results from the AF literature [Rienstra *et al.*, 2020] to our setting. We so obtain that our independence notion for ABA is a semi-graphoid.

Proposition 4.5. *σ-independence in ABA is a semi-graphoid.*

Since each AF induces an ABAF [Toni, 2012] we can transfer ABA independence results to AFs as well.

Proposition 4.6. *Let $F = (\mathbb{A}, \mathbb{R})$ be an AF, D_F its associated ABAF, and $\sigma \in \{co, gr, pr, st\}$. Then, for all $A, B, C \subseteq \mathbb{A}$, we have $A \perp_\sigma B \mid C$ in F iff $A \perp_\sigma B \mid C$ in D_F.*

In what follows, we assume an arbitrary but fixed ABAF $D = (\mathcal{L}, \mathcal{R}, \mathcal{A}, ^-)$. First, note that under gr semantics, all sets are independent since the gr-labelling is unique. Next, we show that two assumption sets are independent when conditioned on a set that uniquely defines one of them.

Definition 4.7. *Assumption set $A \subseteq \mathcal{A}$ σ-defines $B \subseteq \mathcal{A}$ iff for all partial σ-compatible labellings λ_A of A, there is a unique labelling λ_B which is σ-compatible with D.*

Proposition 4.8. *Let $A, B, C \subseteq \mathcal{A}$ be disjoint sets of assumptions. If C σ-defines B then $A \perp_\sigma B \mid C$.*

As a corollary, we obtain that conditional independence is downwards closed wrt. set inclusion.

Corollary 4.9. *$A \perp_\sigma B \mid C$ implies $A' \perp_\sigma B' \mid C$ for any semantics σ and for all $A' \subseteq A$, $B' \subseteq B$.*

In contrast to d-separation in DAGs (cf. Def. 3.11), it is, however, not possible check independence between sets of assumptions by looking at the pairwise independence of their members. In Example 3.2, for instance, the singletons a, b, c are pairwise st-independent while $\{a, b\}$ and $\{c\}$ are not.

We conclude by showing that deciding independence can be computationally hard. In addition, we identify the complexity of the corresponding decision problem for AFs.

Proposition 4.10. *Let $\mathcal{C}$ denote the class of AFs/flat ABAFs. Deciding σ-independence in $\mathcal{C}$ is Π^P_2-complete for $\sigma \in \{co, st\}$ and Π^P_3-complete for $\sigma = pr$.*

5 SCC-driven Independence Checks

The goal of this section is to establish sound and computationally efficient criteria for checking independence for ABAFs. First, we make use of the close connection between AFs and ABAFs and apply the d-graph method for AFs by [Rienstra *et al.*, 2020] to ABAFs (Section 5.1). It turns out, however, that the applicability of the approach is limited due to the specific structure of the instantiated AF. For the second approach, we first define and investigate *SCC-recursiveness for ABA* based on the dependency graph [Rapberger *et al.*, 2022] in Section 5.2. Next, in Section 5.3, we show that the *SCC-Markov condition* [Rienstra *et al.*, 2020] is satisfied by universal and SCC-recursive ABA semantics, thereby providing the prerequisites for an independence check in ABAFs.

In each of these sections, SCCs will play an important part.

Definition 5.1. *Let $G = (V, E)$ be a directed graph, $N \subseteq V$. N is a* strongly connected component (SCC) *of G iff for all $u, v \in N$ there is a path from u to v and N is $\subseteq$-maximal with this property. $SCC(G)$ denotes the set of all SCCs in G.*

The SCCs of G induce a DAG $C_G = (SCC(G), \{(S,T) \mid S \subseteq PA_G(T)\}$. We call an SCC an *initial SCC* if it has no parents and a *terminal SCC* if it has no children in C_G.

5.1 AF-Based d-Graph Approach for ABA

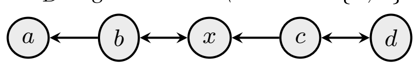
Rienstra *et al.* [2020] develop a sound method to check independence in polynomial time: they transform a given AF into a DAG where each node corresponds to an argument or an SCC and, for a given SCC node, outgoing arcs encode membership and incoming arcs its parents in the AF. We restate their definition of the *d-graph* below.

Definition 5.2. *Given a directed graph $Q = (V, E)$, the* d-graph *$G_Q = (V_d, E_d)$ is a DAG with $V_d = V \cup \{s \mid S \in SCC(F)\}$; and $(u, v) \in E_d$ iff (i) $v \in V$, $u \in SCC(F)$ and $v \in u$ or (ii) $u \in V$, $v \in SCC(F)$ and $u \in PA_G(v)$.*

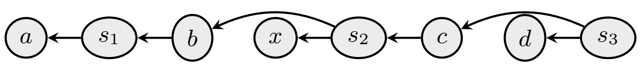
This seems promising at first sight: by Proposition 4.3, we can instantiate an ABAF and use the d-graph approach on the AF to check for independence in ABAFs. Below, we identify assumptions with their corresponding nodes in the d-graph.

Theorem 5.3. *Let $D = (\mathcal{L}, \mathcal{R}, \mathcal{A}, ^-)$ be an ABAF, F_D the corresponding AF, G_{F_D} the d-graph resulting from F_D, and let $\sigma \in \{pr, co\}$. Then, for disjoint sets $A, B, C \subseteq \mathcal{A}$ of assumptions, $A \perp_d B \mid C$ in G_{F_D} implies $A \perp_\sigma B \mid C$ in D.*

Example 5.4. *Let $D = (\mathcal{L}, \mathcal{L}, \mathcal{R}, ^-)$ with $\mathcal{A} = \{a, b, c, d\}$, contraries $\overline{a} = b$, $\overline{d} = c$, $\overline{c} = d$, and rule ($\overline{b} \leftarrow a, d$). The resulting AF F_D is given below (with $x = \{a, d\} \vdash \overline{b}$):*



We have three SCCs: $S_1 = \{a\}$, $S_2 = \{b, x\}$, $S_3 = \{c, d\}$. From the d-graph approach, we obtain the following DAG:



We can use the DAG to identify independencies between assumptions: for given sets of assumptions A, B, C, check if C d-separates A and B; e.g., since b cuts the path between a and c, we infer $\{a\} \perp_\sigma \{c\} \mid \{d\}$, $\sigma \in \{pr, co\}$, in the ABAF.

To avoid the exponential blow-up of the AF instantiation, we can utilize the poly-sized, polynomial-time computable, and (under projection) semantics-preserving *AF-sensitive instantiation* ([Lehtonen *et al.*, 2023], cf. Def. D.5)

Proposition 5.5. *Checking independence in ABA using the d-graph approach on its AF-sensitive instantiation is in* P.

Note that we require an SCC-structure which is informative enough. The SCCs that contain assumption arguments are however often either terminal or initial SCCs (cf. Proposition D.2). In an ABAF with separated contraries, i. e., if $\mathcal{A} \cap \overline{\mathcal{A}} = \emptyset$, conditioning on a set C cannot make assumption sets independent if they are not independent already.

Proposition 5.6. *Let $D = (\mathcal{L}, \mathcal{A}, \mathcal{R}, ^-)$ be an ABAF, $\mathcal{A} \cap \overline{\mathcal{A}} = \emptyset$, F_D its corresponding AF, and G_{F_D} the resulting d-graph, Then, for all disjoint sets $A, B, C \subseteq \mathcal{A}$, if $A \not\perp_{I_D} B \mid C$ in G_{F_D} then $A \not\perp_d B \mid C'$ in G_{F_D} for all assumption sets $C' \supseteq C$.*

Indeed, in case of our initial example not much can be said.

Example 5.7. *In the associated AF F_D of the ABAF D from Example 4.2, all assumptions are terminal SCCs. When applying the d-graph approach, we get a DAG where all nodes that correspond to assumptions are connected with collider-free paths. Also, we cannot use assumptions to d-separate them; thus, $A \not\perp_d B \mid C$ for all assumption sets $A, B, C \subseteq \mathcal{A}$.*

In the next two subsections, we propose a native SCC-based method for ABAFs that circumvents the AF instantiation by exploiting the attack structure of the ABAF directly.

5.2 SCC-Recursiveness for ABA

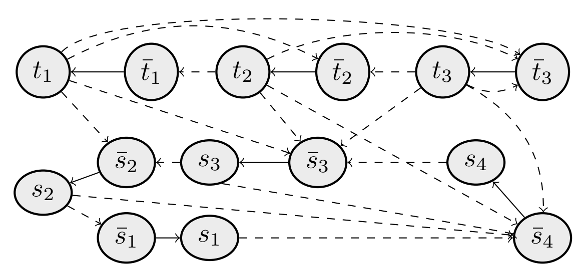
We explore the notion of *SCC-recursiveness* in ABA. In our context, we first need to clarify what the SCCs of an ABAF are. For this, we make use of the dependency graph [Rapberger *et al.*, 2022].

Definition 5.8. *The* dependency graph *$P_D = (V, E, l)$ for an ABAF $D = (\mathcal{L}, \mathcal{R}, \mathcal{A}, ^-)$ is an edge-labelled graph with $V = \mathcal{L}$; and edge $e = (s, t) \in E$ iff (i) there is some rule $r \in \mathcal{R}$ with $s \in body(r)$ and $head(r) = \{t\}$, in this case $l(e) = +$; or (ii) $t \in \mathcal{A}$ and $\overline{t} = s$, in this case, $l(e) = -$.*

Example 5.9. *Let $D = (\mathcal{L}, \mathcal{R}, \mathcal{A}, ^-)$ be an ABAF with $\mathcal{A} = \{t_1, t_2, t_3, s_1, s_2, s_3, s_4\}$, and the following rules:*

$$\begin{array}{lllll} \bar{t}_1 \leftarrow t_2 & \bar{t}_2 \leftarrow t_1 & \bar{t}_2 \leftarrow t_3 & \bar{t}_3 \leftarrow t_1, t_2 & \bar{t}_3 \leftarrow t_3 \\ \bar{s}_1 \leftarrow s_2 & \bar{s}_2 \leftarrow s_3 & \bar{s}_2 \leftarrow t_1 & \bar{s}_3 \leftarrow t_1, t_2 & \bar{s}_3 \leftarrow s_4, t_3 \\ & & \bar{s}_4 \leftarrow t_3 & \bar{s}_4 \leftarrow s_3, t_2 & \bar{s}_4 \leftarrow s_1, s_2 \end{array}$$

Then the dependency graph P_D is given below.



Definition 5.10. *Let $D = (\mathcal{L}, \mathcal{R}, \mathcal{A}, ^-)$ be an ABAF with dependency graph P_D. The SCCs of D are given by*

$$SCC(D) = \{N \cap \mathcal{A} \mid N \text{ is SCC of } P_D, N \cap \mathcal{A} \neq \emptyset\}.$$

For $S, T \in SCC(D)$, T is a parent of S iff there is a directed path $T_1 \dots T_n$ of SCCs from T to S in C_{P_D} s.t. $T_1 = T, T_n = S$ and $T_i = \emptyset$ for $i \notin \{1, n\}$.

Let us delve into the concepts needed to understand SCC-recursiveness in ABA. SCC-recursive semantics can be computed locally, starting from the initial SCCs. In what follows, we will go through such a recursion step by step, using Example 5.9, and introduce all required concepts along the way.

Example 5.11 (Ex. 5.9, cont.). *Our ABAF D from before has two SCCs: $S = \{s_1, s_2, s_3, s_4\}$ and $T = \{t_1, t_2, t_3\}$. Here, T is an initial SCC, the (unique) parent SCC of S.*

In the first step, we compute all σ-labellings for the single initial SCC T. If a given SCC has no parents, the computation amounts to evoking a so-called base function *that returns all* *its σ-labellings. Let $\sigma = co$. We obtain two labellings λ_1 and λ_2 with $\lambda_1(t_i) = \texttt{und}$, $i \leq 3$, and $\lambda_2(t_1) = \texttt{in}$, $\lambda_2(t_2) = \texttt{out}$, $\lambda_2(t_3) = \texttt{und}$. We continue with labelling $\lambda = \lambda_2$.*

When computing the labellings for a non-initial SCC, the first step is to partition its elements based on the incoming arcs of the parent SCCs. In contrast to AFs, we can have attacks where the attacking assumption set consists both of assumptions from the parent and the current SCC, similar as in SCC-recursiveness for SETAFs [Nielsen and Parsons, 2006], cf. [König *et al.*, 2022; Dvorák *et al.*, 2024]. Before coming up with a suitable partition of the assumptions in a given SCC, we will first partition the rules accordingly.

Definition 5.12. *Let $D = (\mathcal{L}, \mathcal{R}, \mathcal{A}, ^-)$ be an ABAF, $A \subseteq \mathcal{A}$ an assumption set and $\lambda : A \to L$ a labelling. A rule $r \in \mathcal{R}$ is*

deactivated, $r \in \mathcal{R}_d(\lambda)$ *iff* $\lambda(a_i) = \texttt{out}$ *for some assumption* $a_i \in body(r) \cap A$

semi-active, $r \in \mathcal{R}_s(\lambda)$ *iff* $\lambda(a_i) = \texttt{und}$ *for some assumption* $a_i \in body(r) \cap A$ *and* $r \notin \mathcal{R}_d(\lambda)$

active, $r \in \mathcal{R}_a(\lambda)$ *iff r is neither deactivated nor semi-active.*

We omit λ and write $\mathcal{R}_s, \mathcal{R}_a$ if it does not cause confusion.

Example 5.13 (Ex. 5.9, cont.). *We now want to move on and compute S, based on the information we got from T. Labelling λ deactivates rules with (the $\texttt{out}$-labelled) t_2 in the body (first row); it renders non-deactivated rules using t_3 semi-active (second row); the rest is active (last row).*

$$\begin{array}{lllll} \bar{t}_1 \leftarrow t_2 & \bar{t}_3 \leftarrow t_1, t_2 & \bar{s}_3 \leftarrow t_1, t_2 & \bar{s}_4 \leftarrow s_3, t_2 & \\ \bar{t}_3 \leftarrow t_3 & \bar{t}_2 \leftarrow t_3 & \bar{s}_3 \leftarrow s_4, t_3 & \bar{s}_4 \leftarrow t_3 & \\ \bar{t}_2 \leftarrow t_1 & \bar{s}_1 \leftarrow s_2 & \bar{s}_2 \leftarrow s_3 & \bar{s}_4 \leftarrow s_1, s_2 & \bar{s}_2 \leftarrow t_1 \end{array}$$

We are ready to define the partition of S into *defeated*, *provisionally defeated*, and *undefeated* assumptions.

Definition 5.14. *Let $D = (\mathcal{L}, \mathcal{R}, \mathcal{A}, ^-)$ be an ABAF, $S \in SCC(D)$, $\lambda : PA_D(S) \to L$ a labelling, and $\mathcal{R}_a, \mathcal{R}_s$ sets of active and semi-active rules. An assumption $a \in S$ is*

defeated, $a \in def_D(S, \lambda, \mathcal{R}_a, \mathcal{R}_s)$ *iff* $\texttt{in}(\lambda) \vdash_{\mathcal{R}_a} \overline{a}$

provisionally defeated, $a \in prdef_D(S, \lambda, \mathcal{R}_a, \mathcal{R}_s)$ *iff* a *is not defeated and* $\mathcal{A} \setminus (S \cup \texttt{out}(\lambda)) \vdash_{\mathcal{R}_a \cup \mathcal{R}_s} \overline{a}$

undefeated, $a \in undef_D(S, \lambda, \mathcal{R}_a, \mathcal{R}_s)$ *iff a is neither defeated nor provisionally defeated*

These sets can be considered a pre-selection for a labelling on S. We define a provisional labelling λ_p on $S \cup PA_D(S)$.

Definition 5.15. *Let $D = (\mathcal{L}, \mathcal{R}, \mathcal{A}, ^-)$ be an ABAF, $S \in SCC(D)$, $\lambda : PA_D(S) \to L$ a labelling, and $\mathcal{R}_a, \mathcal{R}_s$ sets of (semi-)active rules. We define the partial provisional labelling $\lambda_p(S, \lambda, \mathcal{R}_a, \mathcal{R}_s) : S \cup PA_D(S) \to L$ by*

$$\texttt{out}(\lambda_p) = def_D(S, \lambda, \mathcal{R}_a, \mathcal{R}_s) \cup \texttt{out}(\lambda)$$
$$\texttt{und}(\lambda_p) = prdef_D(S, \lambda, \mathcal{R}_a, \mathcal{R}_s) \cup \texttt{und}(\lambda)$$
$$\texttt{in}(\lambda_p) = undef_D(S, \lambda, \mathcal{R}_a, \mathcal{R}_s) \cup \texttt{in}(\lambda)$$

Example 5.16 (Ex. 5.9, cont.). *In D, the assumption s_2 is defeated, s_4 is provisionally defeated, and $\{s_1, s_3\}$ are undefeated. We use this partition of S to extend λ: the so-obtained provisional labelling λ_p is given by $\texttt{out}(\lambda_p) = \{s_2, t_2\}$, $\texttt{und}(\lambda_p) = \{s_4, t_3\}$, and $\texttt{in}(\lambda_p) = \{t_1, s_1, s_3\}$.*

To carry out the recursion on the given SCC S, we restrict our ABAF using λ_p. The restriction $D{\downarrow}_S^{\lambda_p}$ contains only the provisionally defeated and undefeated assumptions of S; to compute our new rule set, we first update $\mathcal{R}_a$ and $\mathcal{R}_s$ to get rid of rules that use defeated assumptions from S.

Definition 5.17. *Let $D = (\mathcal{L}, \mathcal{R}, \mathcal{A}, ^-)$ be an ABAF and $\lambda_p(S, \lambda, \mathcal{R}_a, \mathcal{R}_s)$ a provisional labelling for an SCC $S \in SCC(D)$, and (semi-)active rule sets $\mathcal{R}_a^{up}, \mathcal{R}_s^{up}$. We define*

- *the* restriction $D{\downarrow}_S^{\lambda_p} = (\mathcal{L}, \mathcal{R}', \mathcal{A}', ^-)$ *of D to S under λ_p,*

  $\mathcal{A}' = S \cap (\texttt{in}(\lambda_p) \cup \texttt{und}(\lambda_p))$
  $\mathcal{R}' = \{head(r) \leftarrow body(r) \setminus (\mathcal{A} \setminus S) \mid r \in \mathcal{R}_a^{up} \cup \mathcal{R}_s^{up}\}$

  where $\mathcal{R}_s^{up} = ((\mathcal{R}_a(\lambda_p) \setminus \mathcal{R}_a^{up}) \cup \mathcal{R}_s(\lambda_p)) \cap (\mathcal{R}_a \cup \mathcal{R}_s)$ and $\mathcal{R}_a^{up} = \mathcal{R}_a(\lambda_p) \cap \mathcal{R}_a$ are the updated *sets of semi-active and active rules, respectively; and*
- *the* restricted rule sets *to S, λ_p are $\mathcal{R}_s{\downarrow}_S^{\lambda_p} = \mathcal{R}' \setminus \mathcal{R}_a{\downarrow}_S^{\lambda_p}$ and $\mathcal{R}_a{\downarrow}_S^{\lambda_p} = \{head(r) \leftarrow body(r) \setminus (\mathcal{A} \setminus S) \mid r \in \mathcal{R}_a^{up}\}$.*

Example 5.18 (Ex. 5.9, cont.). *Restriction $D{\downarrow}_S^{\lambda_p}$ has assumptions s_1, s_3, s_4. To compute $\mathcal{R}'$, we first update our rule sets:*

$$\begin{array}{lllll} \mathcal{R}_s^{up}: & \bar{t}_3 \leftarrow t_3 & \bar{t}_2 \leftarrow t_3 & \bar{s}_3 \leftarrow s_4, t_3 & \bar{s}_4 \leftarrow t_3 \\ \mathcal{R}_a^{up}: & \bar{t}_2 \leftarrow t_1 & \bar{s}_2 \leftarrow s_3 & \bar{s}_2 \leftarrow t_1 & \end{array}$$

Now, $\mathcal{R}'$ contains the updated (semi-)active rules where all for the restriction irrelevant assumptions, i. e., assumptions not contained in $\{s_1, s_3, s_4\}$, are removed from the bodies:

$$\begin{array}{rllll} \text{semi-active in } S: & \bar{t}_3 \leftarrow & \bar{t}_2 \leftarrow & \bar{s}_3 \leftarrow s_4 & \bar{s}_4 \leftarrow \\ \text{active in } S: & \bar{t}_2 \leftarrow & \bar{s}_2 \leftarrow s_3 & \bar{s}_2 \leftarrow & \end{array}$$

Now, without further knowledge about which of the rules are semi-active only, we could come to the false conclusion that s_3 can be accepted because s_4 is attacked by a fact (i. e., by $\emptyset$). However, the rule ($s_4 \leftarrow$) is semi-active because it had an $\texttt{und}$-labelled assumption in its body, meaning that it cannot be counted as a successful attack against s_4. We thus remember the restricted rule sets $\mathcal{R}_s{\downarrow}_S^{\lambda_p}$ and $\mathcal{R}_a{\downarrow}_S^{\lambda_p}$ (corresp. to the semi-active resp. active rules in S) for the next recursion step.

$D{\downarrow}_S^{\lambda_p}$ has three SCCs: $S_1 = \{s_1\}$, $S_2 = \{s_3\}$, $S_3 = \{s_4\}$. Again, we start with the initial SCCs. Taking $\mathcal{R}_s$ into account, we obtain label $\texttt{und}$ for s_3 and s_4, $\texttt{in}$ for s_1, $\texttt{out}$ for s_2. Together with λ, we thus get a co-labelling for D.

We are ready to state our definition for SCC-recursiveness. We use a function $GF(D, E, \mathcal{R}_a, \mathcal{R}_s)$, that takes as parameters an ABAF D, a set of candidate assumptions E, sets of (semi-)active rules $\mathcal{R}_s$, $\mathcal{R}_a$ to compute a set of labellings on D. A semantics σ is SCC-recursive if GF can be computed locally on each SCC with a σ-specific base function BF.

Definition 5.19. *A semantics σ satisfies SCC-recursiveness if there exists a local function $BF_\sigma(D, E, \mathcal{R}_a, \mathcal{R}_s)$ such that for every ABAF $D = (\mathcal{L}, \mathcal{R}, \mathcal{A}, ^-)$ we have $\lambda \in \Lambda_\sigma(D)$ iff $\lambda \in GF_\sigma(D, \mathcal{A}, \mathcal{R}, \emptyset)$ where $\lambda \in GF_\sigma(D, E, \mathcal{R}_a, \mathcal{R}_s)$ iff*

- *if $|SCC(D)| = 1$, $\lambda \in BF_\sigma(D, E, \mathcal{R}_a, \mathcal{R}_s)$*

- *otherwise, for each $S \in SCC(D)$, $\lambda|_S = \lambda'$ for some $\lambda' \in GF_\sigma(D{\downarrow}_S^{\lambda_p}, \mathtt{in}(\lambda_p), \mathcal{R}_a{\downarrow}_S^{\lambda_p}, \mathcal{R}_s{\downarrow}_S^{\lambda_p})$ on $D{\downarrow}_S^{\lambda_p}$, and $\lambda(s) = \mathtt{out}$ for $s \in S \setminus D{\downarrow}_S^{\lambda_p}$, for the partial provisional labelling $\lambda_p(S, \lambda|_{PA_D(S)}, \mathcal{R}_a, \mathcal{R}_s)$.*

We give our main result of this subsection. The σ-specific base functions are given at the end of Appendix E.

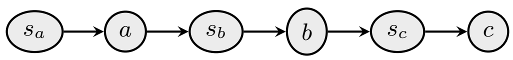
Theorem 5.20. *Each ABA semantics $\sigma \in \{co, pr, st, gr\}$ satisfies SCC-recursiveness.*

5.3 Direct d-Graph Approach for ABA

We propose a sound method to determine conditional independence that does not rely on the AF instantiation. The idea is to apply the d-graph approach to the SCCs of the dependency graph of an ABAF instead of the associated AF.

Definition 5.21. *The* d-graph *of an ABAF $D = (\mathcal{L}, \mathcal{R}, \mathcal{A}, ^-)$ is $G_D = G_{P_D}$ the d-graph wrt. its dependency graph P_D.*

Example 5.22. *Consider an ABAF D with assumptions a, b, c and rules $(\overline{a} \leftarrow a), (\overline{b} \leftarrow a), (\overline{c} \leftarrow b)$. Each assumption has its own SCC in P_D, thus, G_D is a simple chain:*



For the check, we now first compute the dependency graph P_D of the ABAF D, then build the d-graph G_D based on the SCCs of P_D, and, finally, use the d-separation criterion in G_D to check independence between sets of assumptions. To show soundness of the approach, we recall the crucial SCC-Markov property [Rienstra *et al.*, 2020], which generalizes the local Markov property (cf. Def. A.4).

Definition 5.23. *A dependency model I satisfies the* SCC-Markov condition *wrt. graph G iff, for all $S \in SCC(G)$, $S \perp_I ND_G(S) \mid PA_G(S)$.*

We call σ an SCC-Markovian semantics if I_σ satisfies the SCC-Markov condition wrt. P_D. To guarantee this property, we require σ to be SCC-recursive, universal ($\Lambda_\sigma(D) \neq \emptyset$ for all D), and admissible ($\mathtt{in}(\lambda)$ is admissible for all $\lambda \in \Lambda_\sigma(D)$).

We provide a counterpart of [Rienstra *et al.*, 2020, Thm. 2].

Theorem 5.24. *If σ is SCC-recursive, admissible, and universal, then I_σ satisfies the SCC-Markov condition.*

Since co, gr, and pr are admissible, universal, and, by Theorem 5.20, also SCC-recursive, we conclude the following.

Corollary 5.25. *Each $\sigma \in \{co, pr, gr\}$ is SCC-Markovian.*

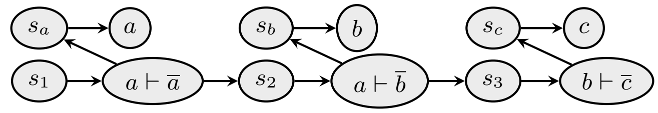
We show that the native ABA d-graph approach is sound: we can indeed use the d-separation criterion in d-graph G_D to check independence for SCC-Markovian ABA semantics.

Theorem 5.26. *Let $D = (\mathcal{L}, \mathcal{R}, \mathcal{A}, ^-)$ be an ABAF and σ be SCC-Markovian. Then, for disjoint sets $A, B, C \subseteq \mathcal{A}$ of assumptions, $A \perp_d B \mid C$ in G_D implies $A \perp_\sigma B \mid C$ in D.*

The approach is in P since constructing P_D is linear in the size of D. Thus, we can check for σ-independence between assumptions in polynomial time for all except st semantics.

As it turns out, the native d-graph approach for ABAFs reveals different independencies than the AF-approach.

Example 5.27. *All assumptions of the d-graph G_{F_D} of AF F_D for ABAF D from Example 5.22 are in terminal SCCs.*



In G_{F_D}, we cannot say if a and c are independent, given b, since they are connected by a collider-free path not containing b. In contrast, b separates a and c in G_D depicted in Example 5.22. So, from G_D we can conclude $a \perp_\sigma c \mid b$ for all semantics σ which satisfy the conditions in Theorem 5.24.

As for AFs, all elements in the same SCC are dependent. As a result, we can only determine independence between assumptions from different SCCs. Thus, just like the AF-approach, the native ABA-approach is incomplete, i.e., not all independencies can be identified. That being said, the ABA d-graph approach still gives us information the AF-approach cannot procure, specifically for ABAFs with separated contraries. Finally, let us head back to our introductory example.

Example 5.28. *In the d-graph G_D for the ABAF D from Example 3.2, assumption d separates every undirected path from w to, e.g., a. Therefore $a \perp_\sigma w \mid d$ in D for $\sigma \in \{co, pr, gr\}$.*

6 Summary and Conclusion

We have conducted a comprehensive study of a semantical notion for conditional independence between assumptions in flat ABA, including satisfaction of semi-graphoid axioms, results on computational complexity and two methods for a sound check of independence in polynomial time, the latter based on our novel notion of SCC-recursiveness for ABA. The proposed independence model is based on three-valued labellings, a choice which was motivated by promising results for a similar notion in the setting of abstract argumentation [Rienstra *et al.*, 2020]. Other concepts of independence between assumptions are feasible, though. In particular, defining independence wrt. joint acceptability of assumptions instead proved to be a well-behaved notion in a preliminary analysis which is beyond the scope of this paper. Diametric to this future work direction, ABA frameworks offer rich structural aspects, and investigating independence between literals, rules and independence in dynamic settings, also for non-flat ABA, are natural next steps. Specifically, our approach could empower enforcement methods as well as repairing and forgetting research [Rapberger and Ulbricht, 2023; Rapberger and Ulbricht, 2024; Berthold *et al.*, 2023] by enabling the decomposition of global reasoning tasks into local reasoning tasks in smaller instances. The high computational complexity of deciding independence may prove challenging for these studies. Overcoming their inherent limitations when it comes to handling ABAFs with large SCCs we consider an interesting albeit possibly challenging research objective. Lastly, ABA is closely related to other forms of structured argumentation, the adaptation of our results to other structured argumentation formalisms such as ASPIC$^+$ [Modgil and Prakken, 2013] or defeasible logic programming [García and Simari, 2004] would be worth investigating.

Ethical Statement

There are no ethical issues.

Acknowledgments

The research reported here was partially supported by the Deutsche Forschungsgemeinschaft (grant 550735820), by ERC under the EU's Horizon 2020 research and innovation programme (grant agreement no. 101020934, ADIX), and by J.P. Morgan and the Royal Academy of Engineering under the Research Chairs and Senior Research Fellowships scheme (grant agreement no. RCSRF2021\11\45).

References

[Belle *et al.*, 2024] Vaishak Belle, Hana Chockler, Shannon Vallor, Kush R. Varshney, Joost Vennekens, and Sander Beckers. Trustworthiness and responsibility in AI - causality, learning, and verification (dagstuhl seminar 24121). *Dagstuhl Reports*, 14(3):75–91, 2024.

[Berthold *et al.*, 2023] Matti Berthold, Anna Rapberger, and Markus Ulbricht. Forgetting aspects in assumption-based argumentation. In *Proceedings of the 20th International Conference on Principles of Knowledge Representation and Reasoning (KR 2023)*, pages 86–96, 2023.

[Besnard *et al.*, 2014a] Philippe Besnard, Marie-Odile Cordier, and Yves Moinard. Arguments using ontological and causal knowledge. In *Foundations of Information and Knowledge Systems - 8th International Symposium (FoIKS 2014)*, volume 8367 of *LNCS*, pages 79–96. Springer, 2014.

[Besnard *et al.*, 2014b] Philippe Besnard, Alejandro Javier García, Anthony Hunter, Sanjay Modgil, Henry Prakken, Guillermo Ricardo Simari, and Francesca Toni. Introduction to structured argumentation. *Argument Comput.*, 5(1):1–4, 2014.

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

[Caminada and Pigozzi, 2011] Martin Caminada and Gabriella Pigozzi. On judgment aggregation in abstract argumentation. *Auton. Agents Multi Agent Syst.*, 22(1):64–102, 2011.

[Craven and Toni, 2016] Robert Craven and Francesca Toni. Argument graphs and assumption-based argumentation. *Artif. Intell.*, 233:1–59, 2016.

[Čyras *et al.*, 2018] Kristijonas Čyras, Xiuyi Fan, Claudia Schulz, and Francesca Toni. Assumption-based argumentation: Disputes, explanations, preferences. In *Handbook of Formal Argumentation*, chapter 7, pages 365–408. College Publications, 2018.

[Cyras *et al.*, 2021a] Kristijonas Cyras, Quentin Heinrich, and Francesca Toni. Computational complexity of flat and generic assumption-based argumentation, with and without probabilities. *Artif. Intell.*, 293:103449, 2021.

[Cyras *et al.*, 2021b] Kristijonas Cyras, Tiago Oliveira, Amin Karamlou, and Francesca Toni. Assumption-based argumentation with preferences and goals for patient-centric reasoning with interacting clinical guidelines. *Argument Comput.*, 12(2):149–189, 2021.

[Darwiche and Pearl, 1994] Adnan Darwiche and Judea Pearl. Symbolic causal networks. In *Proceedings of the 12th National Conference on Artificial Intelligence, Seattle (AAAI 1994)*, pages 238–244. AAAI Press / The MIT Press, 1994.

[Darwiche, 1997] Adnan Darwiche. A logical notion of conditional independence: properties and applications. *Artif. Intell.*, 97(1):45–82, 1997.

[Darwiche, 2009] Adnan Darwiche. *Modeling and Reasoning with Bayesian Networks*. Cambridge University Press, 2009.

[Dung, 1995] Phan Minh Dung. On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games. *Artif. Intell.*, 77(2):321–357, 1995.

[Dvorák *et al.*, 2024] Wolfgang Dvorák, Matthias König, Markus Ulbricht, and Stefan Woltran. Principles and their computational consequences for argumentation frameworks with collective attacks. *J. Artif. Intell. Res.*, 79:69–136, 2024.

[Fan, 2018] Xiuyi Fan. On generating explainable plans with assumption-based argumentation. In *Principles and Practice of Multi-Agent Systems (PRIMA 2018)*, volume 11224 of *LNCS*, pages 344–361. Springer, 2018.

[García and Simari, 2004] Alejandro Javier García and Guillermo Ricardo Simari. Defeasible logic programming: An argumentative approach. *Theory Pract. Log. Program.*, 4(1-2):95–138, 2004.

[Geiger *et al.*, 1990] Dan Geiger, Thomas Verma, and Judea Pearl. Identifying independence in bayesian networks. *Networks*, 20(5):507–534, 1990.

[Halpern and Pearl, 2001] Joseph Y. Halpern and Judea Pearl. Causes and explanations: A structural-model approach: Part 1: Causes. In *Proceedings of the 17th Conference in Uncertainty in Artificial Intelligence (UAI 2001)*, pages 194–202. Morgan Kaufmann, 2001.

[Heyninck, 2023] Jesse Heyninck. An algebraic notion of conditional independence, and its application to knowledge representation (preliminary report). In *Proceedings of the 21st International Workshop on Non-Monotonic Reasoning (NMR 2023)*, volume 3464 of *CEUR Workshop Proceedings*, pages 64–73. CEUR-WS.org, 2023.

[König *et al.*, 2022] Matthias König, Anna Rapberger, and Markus Ulbricht. Just a matter of perspective. In *Computational Models of Argument - Proceedings of COMMA 2022*, volume 353 of *FAIA*, pages 212–223. IOS Press, 2022.

[Lang *et al.*, 2002] Jérôme Lang, Paolo Liberatore, and Pierre Marquis. Conditional independence in propositional logic. *Artif. Intell.*, 141(1/2):79–121, 2002.

[Lehtonen *et al.*, 2023] Tuomo Lehtonen, Anna Rapberger, Markus Ulbricht, and Johannes Peter Wallner. Argumentation frameworks induced by assumption-based argumentation: Relating size and complexity. In *Proceedings of the 20th International Conference on Principles of Knowledge Representation and Reasoning (KR 2023)*, pages 440–450, 2023.

[Lehtonen *et al.*, 2024a] Tuomo Lehtonen, Anna Rapberger, Francesca Toni, Markus Ulbricht, and Johannes Peter Wallner. Instantiations and computational aspects of non-flat assumption-based argumentation. In *Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence (IJCAI 2024)*, pages 3457–3465. ijcai.org, 2024.

[Lehtonen *et al.*, 2024b] Tuomo Lehtonen, Anna Rapberger, Francesca Toni, Markus Ulbricht, and Johannes Peter Wallner. On computing admissibility in ABA. In *Computational Models of Argument - Proceedings of COMMA 2024*, volume 388 of *FAIA*, pages 121–132. IOS Press, 2024.

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

[Nielsen and Parsons, 2006] Søren Holbech Nielsen and Simon Parsons. A generalization of Dung's abstract framework for argumentation: Arguing with sets of attacking arguments. In *ArgMAS 2006, Proceedings*, volume 4766 of *LNCS*, pages 54–73. Springer, 2006.

[Pearl and Paz, 1986] Judea Pearl and Azaria Paz. Graphoids: Graph-based logic for reasoning about relevance relations or when would x tell you more about y if you already know z? In *Proceedings of the 17th European Conference on Artificial Intelligence (ECAI 1986)*, pages 357–363. North-Holland, 1986.

[Pearl, 2009] Judea Pearl. *Causality: Models, Reasoning and Inference*. Cambridge University Press, 2009. 2nd Edition.

[Rapberger and Ulbricht, 2023] Anna Rapberger and Markus Ulbricht. On dynamics in structured argumentation formalisms. *J. Artif. Intell. Res.*, 77:563–643, 2023.

[Rapberger and Ulbricht, 2024] Anna Rapberger and Markus Ulbricht. Repairing assumption-based argumentation frameworks. In *Proceedings of the 21st International Conference on Principles of Knowledge Representation and Reasoning (KR 2024)*, 2024.

[Rapberger *et al.*, 2022] Anna Rapberger, Markus Ulbricht, and Johannes Peter Wallner. Argumentation frameworks induced by assumption-based argumentation: Relating size and complexity. In *Proceedings of the 20th International Workshop on Non-Monotonic Reasoning (NMR 2022)*, volume 3197 of *CEUR Workshop Proceedings*, pages 92–103. CEUR-WS.org, 2022.

[Rienstra *et al.*, 2020] Tjitze Rienstra, Matthias Thimm, Kristian Kersting, and Xiaoting Shao. Independence and d-separation in abstract argumentation. In *Proceedings of the 17th International Conference on Principles of Knowledge Representation and Reasoning (KR 2020)*, pages 713–722, 2020.

[Russo *et al.*, 2024] Fabrizio Russo, Anna Rapberger, and Francesca Toni. Argumentative causal discovery. In *Proceedings of the 21st International Conference on Principles of Knowledge Representation and Reasoning (KR 2024)*, 2024.

[Schulz and Toni, 2017] Claudia Schulz and Francesca Toni. Labellings for assumption-based and abstract argumentation. *Int. J. Approx. Reason.*, 84:110–149, 2017.

[Toni, 2012] Francesca Toni. Reasoning on the web with assumption-based argumentation. In *Reasoning Web. Semantic Technologies for Advanced Query Answering - 8th International Summer School 2012*, volume 7487 of *LNCS*, pages 370–386. Springer, 2012.

[Toni, 2013] Francesca Toni. A generalised framework for dispute derivations in assumption-based argumentation. *Artif. Intell.*, 195:1–43, 2013.

[Verma and Pearl, 1988] Thomas Verma and Judea Pearl. Causal networks: semantics and expressiveness. In *Proceedings of the 4th Annual Conference on Uncertainty in Artificial Intelligence (UAI 1988)*, pages 69–78. North-Holland, 1988.

[Zhalama *et al.*, 2019] Zhalama, Jiji Zhang, Frederick Eberhardt, Wolfgang Mayer, and Mark Junjie Li. Asp-based discovery of semi-markovian causal models under weaker assumptions. In Sarit Kraus, editor, *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence (IJCAI 2019)*, pages 1488–1494. ijcai.org, 2019.