Improving Reasoning Efficiency in ASPIC+ with Backwards Chaining and Partial Arguments

Hao WU[1,*], Bruno YUN[1] and Nir OREN[1]

[1] *University of Aberdeen, Scotland*

Abstract

The first step in reasoning using an approach such as ASPIC+ requires all arguments to be generated, and doing so is computationally intensive when encountering large or frequently updated defeasible theories. In this paper, we introduce the *informative partial argumentation framework* that enables us to control and customise the argument construction process, potentially improving computational efficiency. In our framework, arguments are constructed in a lazy manner; and through the modularisation of our framework, it is possible to restrict the amount of computation to only the necessary information.

Keywords

Argumentation, Backward-chaining, ASPIC+

1. Introduction

The ASPIC+ framework describes how arguments can be constructed from a knowledge base, and allows for reasoning to take place over the resultant argumentation system. However, generating all arguments is computationally expensive [1, 2], and may not be needed for some forms of reasoning. Constructing arguments which contain only the necessary information for a reasoning task is thus clearly desirable and in this paper, we construct arguments using backwards-chaining to speed up the process of determining a literal's status [2], requiring us to often generate only a subset of arguments.

We begin this paper by describing the ASPIC+ framework and the intuition behind our approach (Sec. 2). Section 3 forms the core of our work, introducing *informative partial arguments* and demonstrating how these can be used to compute the justification status of a literal. In Section 4, we give an overview of factors that affect the efficiency of our approach. In Section 5, we perform an empirical evaluation of our approach and in Section 6, we conclude and discuss future research.

2. Background

We start by providing the necessary definitions for the paper, building on the theoretical framework of Yun et al. [2]. We assume an underlying logical language $\mathscr{L}$ defined over a set of literals

SAFA'22: Fourth International Workshop on Systems and Algorithms for Formal Argumentation 2022, September 13, 2022, Cardiff, Wales, United Kingdom

*Corresponding author.

h.wu.20@abdn.ac.uk (H. WU); bruno.yun@abdn.ac.uk (B. YUN); n.oren@abdn.ac.uk (N. OREN)

which is closed under negation ($\neg$). A rule r takes the form $r : \{\phi_1, ..., \phi_n\} \rightsquigarrow \Phi$, where $n \geq 0$, $\forall i \in \{1, \ldots, n\}, \phi_i \in \mathscr{L}$, $\Phi \in \mathscr{L}$ and $\rightsquigarrow \in \{\rightarrow, \Rightarrow\}$. Intuitively, such a rule encodes the fact that the antecedents $\phi_1, \ldots, \phi_n$ conjunctively yield the conclusion Φ. The symbol $\rightarrow$ indicates that r is strict while $\Rightarrow$ indicates that it is defeasible [3, 4] . We denote the set of all rules as $\mathscr{R}$. We write $a \rightarrow b$ to denote $\{a\} \rightarrow b$ in the case of a single antecedent.

The body of r is the rule's set of antecedents $\{\phi_1, ...\phi_n\}$, and is denoted $body(r)$; $head(r) = \Phi$ is called r's head, while $name(r)$ returns the name of r, which is assumed to be in $\mathscr{L}$. Axioms (resp. ordinary premises) are strict (resp. defeasible) rules with empty bodies. The implication of r is $\rightsquigarrow \in \{\Rightarrow, \rightarrow\}$ and is denoted $imp(r)$. A defeasible theory $\mathscr{T}$ is a pair $(\mathscr{S}, \mathscr{D})$ where $\mathscr{S} \subseteq \mathscr{R}$ is a set of strict rules and $\mathscr{D} \subseteq \mathscr{R}$ is a set of defeasible rules. Based on the knowledge contained in a given set of rules, we can identify support relations between pairs of propositions as follows.

Definition 1 (Support Between Propositions). *Given $R \subseteq \mathscr{R}$ and a logical language $\mathscr{L}$, a proposition $p \in \mathscr{L}$ is supported by $p' \in \mathscr{L}$, denoted as $p' \overset{R}{\longmapsto} p$, if (1) there exists a rule $r \in R$ such that $head(r) = p$ and $p' \in body(r)$ or (2) there is a sequence of propositions $(p_0, p_1, \ldots, p_n)$ such that for all $0 \leq i \leq n, p_i \in \mathscr{L}$, $p_0 = p'$, $p_n = p$ such that for all $0 \leq i \leq n-1, p_i \overset{R}{\longmapsto} p_{i+1}$.*

It is useful to detect circular supports, that is, a proposition that supports itself given some set of rules. We refer to such a set of rules as *circular* (c.f., [5] in the context of assumption-based argumentation).

Definition 2 (Circular set of rules). *We say that $R \subseteq \mathscr{R}$ is circular iff there is a proposition $p \in \mathscr{L}$ such that $p \overset{R}{\longmapsto} p$.*

We now recall the definition of an argument in ASPIC+ [4].

Definition 3 (ASPIC+ Argument). *Given a defeasible theory $\mathscr{T} = (\mathscr{S}, \mathscr{D})$, an ASPIC+ argument A is either:*

1. *$\emptyset \rightsquigarrow \phi \in \mathscr{S} \cup \mathscr{D}$, $Conc(A) = \phi$, $Sub(A) = \{A\}$, $Prem(A) = \{\phi\}$; or*
2. *$A_1, \ldots, A_n \rightsquigarrow \phi$ where for all $1 \leq i \leq n$, A_i are arguments, $Conc(A) = \phi$, and there exists a rule $\{Conc(A_1), \ldots, Conc(A_n)\} \rightsquigarrow \phi$ in $\mathscr{S} \cup \mathscr{D}$, $Sub(A) = \bigcup_{1 \leq i \leq n} Sub(A_i) \cup \{A\}$, and $Prem(A) = \bigcup_{1 \leq i \leq n} Prem(A_i)$.*

ASPIC+ arguments are generally generated in a bottom-up or forward chaining manner, starting from the axioms or ordinary premises and working forwards towards conclusions. Observe that concepts such as attack or defeat, which are built on arguments, have no effect on the arguments constructed using forward chaining. Clearly, this has the potential to be computationally inefficient as arguments which are known to not affect the literals we are interested in may still be generated. In what follows, we consider techniques for backwards chaining in the argument construction process through the introduction of *informative partial arguments*. Since backwards chaining potentially gives many choice points when selecting what to expand, in the next section, we describe a modular framework where different approaches for expansion selection can be used.

Before that, we introduce some further basic notions and notations about trees. A directed graph is a pair $G = (V, E)$, where V is a set of nodes and $E \subseteq V \times V$ is the set of directed

edges. A *directed path* from node v_0 to node v_n is a sequence $(v_0, v_1, \ldots, v_n)$ such that for all $i \in \{0, \ldots, n-1\}, (v_i, v_{i+1}) \in E$. We say that an acyclic directed graph $G = (V, E)$ is a *directed rooted tree* iff there exists a unique $r \in V$, called the *root* of G such that for all $v' \in V \setminus \{r\}$, there is exactly one directed path from r to v'. Given a directed rooted tree $G = (V, E)$, for all $(v_1, v_2) \in E$ we define $\text{fst}((v_1, v_2)) = v_1$, $\text{snd}((v_1, v_2)) = v_2$. If $(v', v) \in E$, we say that v' is the *parent* of v, denoted as $parent_v$. The node $v \in V$ is a *leaf* of G if $\forall e \in E, v \neq \text{fst}(e)$. The set of all leaves of G is denoted as $LEAVES(G)$.

3. The Informative Partial Argumentation Framework

Our approach for efficiently reasoning within ASPIC+ is based on the concept of backwards chaining which involves expanding a conclusion by selecting some applicable rules (i.e., those whose conclusion is the literal we are hoping to prove). During the expansion process, the objects we are manipulating are not arguments (as they are not rooted in axioms or ordinary premises). We therefore introduce *informative partial arguments* (Definition 11) which include the conclusion of an argument and a subset of the rules necessary to infer it, as well as context information gathered during the expansion. We begin by introducing some basic concepts which are key components of the backward chaining process (Definition 4 to Definition 10). We introduce this process through an illustrative example.

Example 1. *Let us consider $\mathscr{T} = (\mathscr{S}, \mathscr{D})$, where $\mathscr{S} = \{b \to a, a \to b, f \to c, e \to c, \emptyset \to g\}$ and $\mathscr{D} = \{\{c,d\} \Rightarrow b, n \Rightarrow b, g \Rightarrow d, h \Rightarrow f, \emptyset \Rightarrow e, \emptyset \Rightarrow h\}$. Assume that we want to build the arguments that conclude the target proposition a.*

- ***Step 0:*** *At the beginning of the construction, we have only 'a'. We need to find all rules from $\mathscr{T}$ that have head 'a', and observe there is only one such rule $b \to a$. From this rule, we see that there must be a proposition 'b' that supports 'a'.*
- ***Step 1:*** *Now, since we have the rule $b \to a$, we need to find all rules from $\mathscr{T}$ that have head 'b'.There are three such rules, $a \to b$, $\{c,d\} \Rightarrow b$ and $n \Rightarrow b$, these are evidence that b could be minimally supported in three different ways.*
- ***Step 2:*** *We have three sets of rules that indicate our target proposition 'a' could be supported in these different ways, $\{a \to b, b \to a\}$, $\{\{c,d\} \Rightarrow b, b \to a\}$, and $\{n \Rightarrow b, b \to a\}$.*

Although we only highlighted three steps in Example 1, we can make the following observations:

- *The necessary information for backward chaining:* At each step, the input data consists of a set of rules and a set of propositions. Furthermore, it is possible to collect some information during the backward chaining process. For example, we could collect the preference information of defeasible rules as described in [4]. Thus, we argue that some kind of context information should also be included in the backward chaining process.
- *Different Statuses:* In step 2, the set $\{a \to b, b \to a\}$ demonstrates circular reasoning, while the backward chaining process stops at $(\{n \Rightarrow b, b \to a\})$ as there is no evidence in $\mathscr{T}$ that shows that n is supported. The backward chaining process of arguments that conclude a

should be stopped in these two cases, as it will not be possible to form an argument by continuing the process. However, given $\{\{c,d\} \Rightarrow b, b \rightarrow a\}$, there is evidence in $\mathscr{T}$ that indicates that 'c' and 'd' can be supported in different ways, thus, it is possible to continue the backward chaining process in this case.

- *Two Operations:* At each step, we need to both (1) search for rules that can be applied and (2) apply these rules to the current information we have. The composition of these two operations corresponds to our intuition of building an argument backward.

The intuition described in the aforementioned three items are at the heart of the informative partial argument framework. We encode the (abstract) data structure of our framework via *P-structures* which record the necessary information for backward chaining.

Definition 4 (The P-structure). *Given a logical language $\mathscr{L}$, a defeasible theory $\mathscr{T} = (\mathscr{S}, \mathscr{D})$ and an arbitrary set of elements $\mathbb{I}$, a P-structure is a triple $\langle L, R, i\rangle$ where $L \subseteq \mathscr{L}, R \subseteq \mathscr{S} \cup \mathscr{D}$ and $i \in \mathbb{I}$.*

The set $\mathbb{I}$ is used to encode specific contextual information about the backward chaining process. Thus $\mathbb{I}$ may be different if we record the number of rules used in an argument or if we record the set of attacks against an argument. Returning to Example 1, we can encode its data as follows. The initial input at step 0 is a P-structure $\langle\{a\}, \emptyset, i_0\rangle$, the input at step 1 is a P-structure $\langle\{b\}, \{b \rightarrow a\}, i_1\rangle$, and there are three P-structures as inputs at step 2:

- $p_1 = \langle\{a\}, \{a \rightarrow b, b \rightarrow a\}, {i'}_2\rangle$,
- $p_2 = \langle\{c,d\}, \{\{c,d\} \Rightarrow b, b \rightarrow a\}, {i''}_2\rangle$, and
- $p_3 = \langle\{n\}, \{n \Rightarrow b, b \rightarrow a\}, {i'''}_2\rangle$.

The set of literals L in a P-structure contains propositions that are not yet supported, while the set of rules R serves as the evidence that the proposition a are supported.

Definition 5 (Status of a P-structure). *Given a P-structure $\langle L, R, i\rangle$ and a defeasible theory $\mathscr{T} = (\mathscr{S}, \mathscr{D})$, the status of a P-structure is*

- ***Complete** iff $L = \emptyset$.*
- ***Enrichable** iff R is not circular and $\forall \phi \in L$, $\exists r \in (\mathscr{S} \cup \mathscr{D})$ s.t. $head(r) = \phi$.*
- ***Pending** if it is not complete or enrichable.*

At step 2 in Example 1, $p_1 = \langle\{a\}, \{a \rightarrow b, b \rightarrow a\}, {i'}_2\rangle$, and $p_3 = \langle\{n\}, \{n \Rightarrow b, b \rightarrow a\}, {i'''}_2\rangle$ are pending, because the former has a circular set of rules whereas the literal part of the latter has no support from the defeasible theory.

Definition 6 (Support Function). *Given $\mathscr{T} = (\mathscr{S}, \mathscr{D})$, we define a function support$_{\mathscr{T}}$ that takes as input a P-structure $p = \langle L, R, i\rangle$ and returns a set $S \in 2^{(2^{\mathscr{S} \cup \mathscr{D}})}$ such that if p is enrichable then for all $s \in S$, we know that $\{head(r) | r \in s\} = L$, and each $r_1 \in s$, there is no $r_2 \in s$ such that $head(r_1) = head(r_2)$, and S is maximal (w.r.t. set inclusion). If p is not enrichable, then support$_{\mathscr{T}}(p) = \emptyset$.*

In Example 1, given the input data $p_2 = \langle \{c,d\}, \{\{c,d\} \Rightarrow b, b \rightarrow a\}, i''_2 \rangle$, $support_{\mathscr{T}}(p_2) = \{\{f \rightarrow c, g \Rightarrow d\}, \{e \rightarrow c, g \Rightarrow d\}\}$. These two sets indicate that p_2 can be enriched in two different ways.

Next we consider how a P-structure can be transformed to another P-structure. The *link function* takes as input a P-structure, a set of rules and returns a new P-structure that can be obtained by combining all these elements. The link function utilises two other functions: the *control function* that specifies how the status of the new P-structure is controlled and the *extraction function* that specifies how context information is extracted .

Definition 7 (Link Function). *Given a defeasible theory $\mathscr{T} = (\mathscr{S}, \mathscr{D})$, rules $R^e \in 2^{\mathscr{S} \cup \mathscr{D}}$, a set of context information $\mathbb{I}$, a P-structure $p = \langle L, R, i \rangle$, and a binary operator $\bowtie\colon \mathbb{I} \times \mathbb{I} \rightarrow \mathbb{I}$ which specifies how context information is accumulated[1], A link function is a function of the form $\Sigma_{\mathbb{EXTR},\mathbb{CONT},\bowtie}(p, R^e) = \langle L', R', i' \rangle$, where:*

- $L' = \mathbb{CONT}(p, R^e)$ *(see Definition 8),*
- $i' = \mathbb{EXTR}(p, R^e) \bowtie i$ *(see Definition 9), and*
- $R' = R \cup R^e$.

Definition 8 (Control Function). *A control function $\mathbb{CONT}$ takes a P-structure p and a set of rules (evidence returned by* support *function) as inputs and returns a set of literals, i.e., $\mathbb{CONT} : (2^{\mathscr{L}} \times 2^{\mathscr{S} \cup \mathscr{D}} \times \mathbb{I}) \times 2^{\mathscr{S} \cup \mathscr{D}} \rightarrow 2^{\mathscr{L}}$.*

Different instantiations of the *control* function will lead to different P-structures. In the following example, we propose some basic control functions but the framework can easily be expanded to more complex ones without loss of generality.

Example 2. *Given a P-structure $p = \langle L, R, i \rangle$ and $R^e \in 2^{\mathscr{S} \cup \mathscr{D}}$. We provide two examples of control functions, $\mathbb{CONT}_1$ and $\mathbb{CONT}_2$, such that:*

- $\mathbb{CONT}_1(p,\ R^e) = \bigcup_{r \in R^e} body(r)$. *This control function returns all propositions in the bodies of rules in R^e.*
- $\mathbb{CONT}_2(p,\ R^e) = \bigcup_{r \in R^e, r \notin \mathscr{D}} body(r)$. *This control function returns all propositions in the bodies of strict rules in R^e.*

Note that while these two control functions are basic and do not make use of the P-structure p, more complex control functions can be defined.

After applying the control function to a P-structure, we have a set of propositions which require rules to support them within the new P-structure. We must decide which propositions to support so as to obtain new P-structures. Given that many strategies exist for doing so, we may apply different heuristics accordingly. An *extraction* function takes as input a P-structure and returns the information needed by our heuristic. For example, we may need to determine the number (or indeed the set) of all propositions which could be used by the rules supporting a P-structure.

[1] Note that $\mathbb{I}$ can be instantiated in different ways; it is left abstract in this definition. Examples of instantiations of $\mathbb{I}$ are provided in Example 3.

Definition 9 (Extraction Function). *The function* $\mathbb{EXTR}$ *takes a P-structure p and a set of rules (the evidence returned by the* support *function) as input and returns an element of* $\mathbb{I}$*, i.e.,* $\mathbb{EXTR} : (2^{\mathscr{L}} \times 2^{\mathscr{S}\cup\mathscr{D}} \times \mathbb{I}) \times 2^{\mathscr{S}\cup\mathscr{D}} \to \mathbb{I}$.

Example 3. *Given a P-structure* $p = \langle L, R, i\rangle$ *and* $R^e \in 2^{\mathscr{S}\cup\mathscr{D}}$*. We provide two examples of extraction functions,* $\mathbb{EXTR}_1$ *and* $\mathbb{EXTR}_2$*, such that:*

- $\mathbb{EXTR}_1(p,\ R^e) = \{r | r \in R^e, r \in \mathscr{D}\}$*. This extraction function returns all defeasible rules in* R^e*. Here,* $\mathbb{I} = 2^{\mathscr{S}\cup\mathscr{D}}$.
- $\mathbb{EXTR}_2(p,\ R^e) = 2 \times |R \cap \mathscr{S}| - |R^e \cap \mathscr{D}|$*. This extraction function returns 2 times the number of strict rules in R minus the number of defeasible rules in* R^e*. Here,* $\mathbb{I} = \mathbb{N}$.

Using our framework, we can reformulate existing link functions:

- $naiveLink = \Sigma_{\mathbb{EXTR},\mathbb{CONT}_1,\bowtie}$. This link function makes use of $\mathbb{CONT}_1$ (see Example 2) and assume the computational context ignores information $\mathbb{I}$, thus $\mathbb{EXTR}$ and $\bowtie$ can be defined in an arbitrary way.
- $weakestLink = \Sigma_{\mathbb{EXTR}_1,\mathbb{CONT}_1,\cup}$. This link function makes use of $\mathbb{CONT}_1$ (see Example 2), $\mathbb{EXTR}_1$ (see Example 3), and $\bowtie = \cup$.
- $lastLink = \Sigma_{\mathbb{EXTR}_1,\mathbb{CONT}_2,\cup}$. This link function makes use of $\mathbb{CONT}_2$ (see Example 2), $\mathbb{EXTR}_1$ (see Example 3), and $\bowtie = \cup$.

The use of different types of context information ($\mathbb{I}$) will be discussed in Example 4. When definitions of the control/extraction functions and $\bowtie$ are clear from the context, we will simplify $\Sigma_{\mathbb{EXTR},\mathbb{CONT},\bowtie}$ by writing Σ. We can now give the formal definition of the operations described in the steps of Example 1.

Definition 10 (Enrich Function π). *Given* $\mathscr{T} = (\mathscr{S}, \mathscr{D})$*, a P-structure* $p = \langle L, R, i\rangle$*, and a link function* Σ*, the enrich function w.r.t.* $\mathscr{T}$ *and* Σ *is* $\pi_{\mathscr{T},\Sigma} : 2^{\mathscr{L}} \times 2^{\mathscr{S}\cup\mathscr{D}} \times \mathbb{I} \to 2^{(2^{\mathscr{L}} \times 2^{\mathscr{S}\cup\mathscr{D}} \times \mathbb{I})}$ *such that, when* $support_{\mathscr{T}}(p) \neq \emptyset$*:*

$$\pi_{\mathscr{T},\Sigma}(p) = \{\Sigma(p, e) | e \in support_{\mathscr{T}}(p)\}$$

otherwise:

$$\pi_{\mathscr{T},\Sigma}(p) = p$$

The notation $\pi_{\mathscr{T},\Sigma}(p)$ can be written as $\pi(p)$ if $\mathscr{T}$ and Σ are known from the context. Now, all components that are necessary to describe the backward construction of the partial arguments have been formally defined, we can finally introduce the *informative partial arguments* that integrates all these concepts.

Definition 11 (Informative Partial Arguments). *Given* $\mathscr{T} = (\mathscr{S}, \mathscr{D})$*, a link function* Σ*, an initial context information* $i \in \mathbb{I}$*, a target proposition* $\phi \in \mathscr{L}$*, and a natural number* $z \in \mathbb{N}$*,*

we define the directed rooted tree $\mathbb{G}^{\phi}_{z,\Sigma,\mathscr{T}} = (V^{\phi}_z, E^{\phi}_z) = \Pi_{\mathscr{T},\Sigma}((V^{\phi}_{z-1}, E^{\phi}_{z-1}))$*, where* $\mathbb{G}^{\phi}_{0,\Sigma,\mathscr{T}} = (V^{\phi}_0, E^{\phi}_0) = (\{\langle\{\phi\},\emptyset,i\rangle\},\emptyset)$*, and* $\Pi_{\mathscr{T},\Sigma}$ *is defined as:*

$$\Pi_{\mathscr{T},\Sigma}(\mathbb{G}) = ((\bigcup_{v\in LEAVES(\mathbb{G})} \pi(v)) \cup V, \bigcup_{v\in LEAVES(\mathbb{G})} \{(v,v')|v' \in \pi(v)\})$$

Symbols Σ *and* $\mathscr{T}$ *can be omitted when they are known from the context. Each element of* V^{ϕ}_z *is a P-structure, and it is referred to as an* informative partial argument for ϕ*. The root* V^{ϕ}_0 *is the set containing the initial informative partial argument that wraps the target proposition* ϕ *together with the initial context information i.* $\mathbb{G}^{\phi}_z$ *is known as the tree of informative partial argument of* ϕ *at step z, it encodes the history of these z steps backward enrichment process starting from* $\mathbb{G}^{\phi}_0$*. If for all* $v \in LEAVES(\mathbb{G}^{\phi}_{z,\Sigma,\mathscr{T}})$*,* $\pi(v) = \{v\}$ *then we call* $\mathbb{G}^{\phi}_z$ *the terminal tree of informative partial argument for* ϕ*. We denote the terminal tree as* $\mathbb{G}^{\phi}_{\perp} = (V^{\phi}_{\perp}, E^{\phi}_{\perp})$*, and the set* $\{v|v \in V^{\phi}_{\perp},\ v \text{ is complete}\}$ *as* $\mathbb{A}^{\phi}_{\Sigma,\mathscr{T}}$*.*

Example 4 (Cont't Example 1). *To demonstrate how trees of informative partial argument are indexed by different link function instantiations, we define a dummy-link function as followed:*

$$dummyLink = \Sigma_{\mathbb{EXTR}_2,\mathbb{CONT}_1,\mathbb{K}}$$

where for every two natural numbers $x, y \in \mathbb{N}, \mathbb{K}(x,y) = x$*. As shown in Table 1, a* ***tree of informative partial argument*** *for a is indexed by the link function and enrich steps. We choose* $\emptyset$ *and 0 as initial context information respectively.* $\mathbb{G}^{a}_{2,lastLink}$ *is a terminal tree, because* $\langle\{a\},\{a\to b, b\to a\},\emptyset\rangle$ *are of pending status and* $(\langle\emptyset,\{c,d\Rightarrow b, b\to a\},\{c,d\Rightarrow b\}\rangle$ *is complete.*

z	$LEAVES(\mathbb{G}^{a}_{z,lastLink})$	$LEAVES(\mathbb{G}^{a}_{z,dummyLink})$
0	$\langle\{a\},\emptyset,\emptyset\rangle$	$\langle\{a\},\emptyset,0\rangle$
1	$\langle\{b\},\{b\to a\},\emptyset\rangle$	$\langle\{b\},\{b\to a\},0\rangle$
2	$\langle\{a\},\{a\to b, b\to a\},\emptyset\rangle$ $\langle\emptyset,\{c,d\Rightarrow b, b\to a\},\{c,d\Rightarrow b\}\rangle$ $\langle\emptyset,\{n\Rightarrow b, b\to a\},\{n\Rightarrow b\}\rangle$	$\langle\{a\},\{a\to b, b\to a\},2\rangle$ $\langle\{c,d\},\{c,d\Rightarrow b, b\to a\},1\rangle$ $\langle\{n\},\{n\Rightarrow b, b\to a\},1\rangle$
3	Same as above (**Terminated**)	$\langle\{f,g\},\{f\to c, g\Rightarrow d,\{c,d\}\Rightarrow b, b\to a\},1\rangle$ $\langle\{e,g\},\{e\to c, g\Rightarrow d,\{c,d\}\Rightarrow b, b\to a\},1\rangle$

Table 1
Leaves of trees of informative partial arguments for *a* at each step w.r.t. *lastLink* and *dummyLink*.

The motivation of developing this backward construction framework is to identify deductive rules that are equivalent to an ASPIC+ argument efficiently. The relation between an informative partial argument and an ASPIC+ argument is described as follows.

Proposition 1. *Given a defeasible theory* $\mathscr{T}$*,* $\Sigma \in \{naiveLink, weakestLink\}$*, if* $\mathbb{A}^{\alpha}_{\Sigma,\mathscr{T}} \neq \emptyset$*, then, for each* $A = \langle L, R, i\rangle \in \mathbb{A}^{\alpha}_{\Sigma,\mathscr{T}}$*, based on* $\mathscr{T}' = (\mathscr{S}', \mathscr{D}')$*,* $\mathscr{S}' \cup \mathscr{D}' = R$*, there is an unique ASPIC+ argument* A' *that concludes* α*.*

Proof 1. *According to the Definition 5 and Definition 11, we know that there is a natural number z such that* $A = \langle \emptyset, R_z, \_ \rangle \in V_z^{\alpha}$. *As defined in Example 2 and Example 3, naiveLink and weakestLink do not make selection on propositions that need to be supported in the next step, therefore, according to Definition 6, we know that for each* $1 \leq n \leq z-1$, *there exist* $R_n^e \in support_{\mathscr{T}}(v_n)$ *such that* $R_{(n+1)} = R_n^e \cup R_n$ *and* $\{head(r) | r \in R_n^e\} = L_n$. *Given* $A = \{\emptyset, R_z, \_\}$, *for each* $r \in R_{(z-1)}^e, body(r) = \emptyset$, *therefore, each* $r \in R_{z-1}^e$ *is an ASPIC+ argument (the base case of Definition 3) that concludes a unique* $l \in L_{(z-1)}$, *the set of these ASPIC+ arguments is denoted as* $A'_{(z-1)}$. *According to Definition 3 (inductive case (2)) we know that* $A'_{(z-1)}$ *and* $R_{(z-2)}^e$ *forms a set of ASPIC+ argument* $A'_{(z-2)}$ *such that each ASPIC+ argument in* $A'_{(z-2)}$ *concludes a unique* $l \in L_{(z-2)}$. *Similarly, we can say that the set of ASPIC+ arguments* A'_n *and* $R_{(n-1)}^e$ *forms the set of ASPIC+ arguments* $A'_{(n-1)}$, *each ASPIC+ argument in* $A'_{(n-1)}$ *concludes a unique* $l \in L_{(n-1)}$, *and we know* $L_0 = \{\alpha\}$. *So we have* $A'_n \cup \cdots \cup A'_0$ *are all sub arguments of* A'. *Assuming that given* $\mathscr{T}'$, *there are more than one ASPIC+ arguments concludes* α (A'' *and* A'), *it indicates that there are at least two sub-arguments of* A' *and* A'' *respectively, that concludes some proposition* β *(it is possible that* $\beta = \alpha$*), in this case, according to Definition 6,* $R = \mathscr{S}' \cup \mathscr{D}'$ *will lead to more than one informative partial arguments, thus contradict to the assumption.*

Next, we adapt attacks and defeats to our new framework.

Definition 12 (Attack). *Given* $\mathscr{T} = (\mathscr{S}, \mathscr{D})$ *and two trees of informative partial arguments* $\mathbb{G}^{\phi_1}_{z_1,\Sigma_1} = (V_1, E_1)$, $\mathbb{G}^{\phi_2}_{z_2,\Sigma_2} = (V_2, E_2)$ *for* ϕ_1 *and* ϕ_2. *An informative partial argument* $v_1 = \langle L_{v_1}, R_{v_1}, info_{v_1} \rangle \in V_1$ *is attacked by* $v_2 \in V_2$ *if one of the following item holds:*

- $\neg\phi_2 = \phi$ *where* $\phi \in \{head(r) | r \in R_{v_1} \cap \mathscr{D}\}$. *We say* v_2 ***rebuts*** v_1 *on* ϕ.
- $\neg\phi_2 = \phi$ *where* $\phi \in \{name(r) | r \in R_{v_1} \cap \mathscr{D}\}$. *We say* v_2 ***undercuts*** v_1 *on* ϕ.

Proposition ϕ *and* $\phi_2 = \neg\phi$ *are referred as the* ***attacked*** *proposition and the* ***attacking*** *proposition respectively. This definition indicates that attackers rely on attacking propositions only , so* $\mathbb{G}^{\phi_2}_{z_2,\Sigma_2}$ *represents a family of attackers in which each family member is indexed by the choice of* z_2 *and* Σ_2. *Similarly, an argument is attacked only when an attacked proposition is included in its structure, so* $\mathbb{G}^{\phi_1}_{z_1,\Sigma_1}$ *also represents a family of attacked arguments in which each family member is indexed by the choice of* z_1 *and* Σ_1 *such that* $\neg\phi_2$ *is included.*

From Proposition 1, we know that the attack relation that we are familiar with in the ASPIC+ framework [4] is equivalent to a special case that depends on *naiveLink* or *weakestLink*, for example , it can be expressed as $\forall(a,b) \in \mathbb{A}^{\phi_1}_{naiveLink} \times \mathbb{A}^{\phi_2}_{naiveLink}$ such that a attacks b. In order to determine whether a rebut is successful or not, we can use weakest-link or last-link or other link functions to collect context information. So the defeat relation can be formally defined as follows.

Definition 13 (Defeat). *Given* $\mathscr{T} = (\mathscr{S}, \mathscr{D})$, *an ordering relation* $\preceq$ *parameterized by the choice of Info (such as Eli or Dem [4]) and a tree of partial arguments* $\mathbb{G}^{\phi_1}_{z_1,\Sigma_1} = (V, E)$ *for* $\phi_1 \in \mathscr{L}$. *We say an informative partial argument* $v \in V$ *is* Σ_2*-defeated*[2] *by an informative partial argument* b *of* $\phi_2 \in \mathscr{L}$ *when one of following items is satisfied.*

[2]We can also say an informative partial argument b of ϕ_2 Σ_2-defeats v.

- *b undercuts v on* $\neg\phi_2$.
- *b rebuts v on* $\neg\phi_2$, $b \in \mathbb{A}_{\Sigma_2}^{\phi_2}$ *and* $\forall a \in \mathbb{A}_{\Sigma_2}^{\neg\phi_2}$ *we know that* $info_a \preceq info_b$.

To determine if a rebut leads to a successful attack or not, we rely on an instantiation (Σ_2) *of the link function to indicate what information is required for comparison.*

Once a rebut is detected during the enrichment process, it is necessary to collect information specified by Σ_2 from all conflicting arguments. Σ_2 and Σ_1 are not necessarily the same, implying that the objective of constructing an argument could be independent from the process of determining whether or not an argument is defeated.

In next section, we will have a discussion about possible optimisations during the construction process so that we can have a better understanding about the role of our framework.

4. Optimisation options and the scope of heuristic information

We begin this discussion about computation optimisation of higher-level concepts with an example of query answering on a defeasible theory. The purpose of querying a defeasible theory is to determine if a statement is justified by the information contained in it. Being justified could be defined in a variety of ways, such as "there is a naive-complete argument of the query proposition and it is not attacked" or " all naive-complete arguments of the query proposition are not attacked".

Example 5. *Assume that a query proposition is justified if there is a naiveLink-complete argument of this query and it is not* Σ*-defeated by any other argument. Then given* $\mathscr{T} = (\mathscr{S}, \mathscr{D})$*, where* $\mathscr{S} = \{b \to a, \to e, f \to \neg b, \to g, h \to \neg c, \to i, j \to \neg d, \to k\}$ *and* $\mathscr{D} = \{c \Rightarrow b, d \Rightarrow c, e \Rightarrow d, g \Rightarrow f, i \Rightarrow h, k \Rightarrow j\}$*, is 'a' justified and how much computation is necessary for drawing this conclusion?*

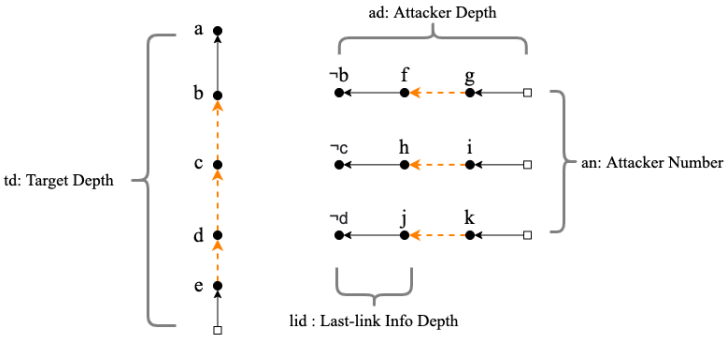

Figure 1: Argument that conclude that *a* is defeated.

The necessary computation for the construction all attackers are constrained by the instantiation of Σ*. When adopting* $\Sigma = lastLink$ *, the axioms in* $\{\to e, \to k, \to j, \to g\}$ *are not necessary for drawing the conclusion of this query whereas with* $\Sigma = weakestLink$*, we would require all rules in* $\mathscr{T}$*.*

In terms of the higher-level structure of the justification criterion in Example 5, there are two optimisation options: (1) Should we compute the $\mathbb{A}^{a}_{naiveLink}$ first then compute the Σ-defeat relation of each attacker; or (2) Should we compute the Σ-defeat relation whenever an attacker is identified at each enrich step toward $\mathbb{A}^{a}_{naiveLink}$?

Option 1 puts us at the risk that the partial argument of proposition $\neg b$ could be a defeater, if $\Sigma = lastLink$ then all computations involving rules $\{d \Rightarrow c, e \Rightarrow d, \rightarrow e\}$ are not necessary. Option 2 puts us at the risk that there is no partial argument of proposition a that is naive-complete. Assume that there is no rule $(\emptyset \rightarrow e)$ in the given defeasible theory in Example 5, then all computation of the Σ-defeat relation would be not necessary.

At the current stage, we have no means of knowing which option is better than the other. Example 5 shows that there are two types of heuristic information: (1) The argument-level heuristic information ($\mathbb{I}$) which has the scope of a path in a tree of informative partial arguments. (2) The defeasible-theory-level heuristic information which has the scope of the entire defeasible theory. Because we do not explore defeasible-theory-level heuristic information in this paper, we will focus on demonstrating the optimization brought about by adopting different link functions in the next section.

5. Performance Evaluation

To emphasise the performance boost offered by our framework alone, we examine the differences between using the *lastLink* and *weakestLink*. The implementation of *weakestLink* is the baseline and could be thought of as a close approximation to the calculation necessary to generate an ASPIC+ argument.

To evaluate the performance with larger size benchmarks and control the complexity, we reuse the justification criterion and dataset structure as shown in Example 5. The proposition a is justified in all benchmarks. The only leaf of $\mathbb{G}^{\phi}_{\Sigma}, \phi \in \mathscr{L}$ is denoted as $leaf(G^{\phi}_{\Sigma})$. The scale of our benchmarks[3] is controlled by three parameters as shown in Figure 1. Here, Target-Depth (td) is the length of the path from the root to $leaf(\mathbb{G}^{a}_{naiveLink})$. The Attacker-Depth ($ad$) is the target-depth of an attacker argument. Finally, the LastLink-Info-Depth (lid) is the length of the path from the root to $leaf(\mathbb{G}^{a}_{\Sigma})$ for a given Σ. There are 10 attackers, and the values of ad and lid of all attackers are the same.

- Benchmark Asymptote: Set td and ad to be control variables and increase the value of lid at a given step [4]. The benchmark includes 99 defeasible theories. As shown in Figure 2, in this case, the length of all attacker arguments is a constant. Adopting *weakestLink* requires the construction of a complete ASPIC+ argument, so the computational performance is relatively stable. Adopting *lastLink* would require only the computation necessary to meet the first defeasible rules, so the computation would take more time as the depth of the first defeasible increased, and in the worst case it will achieve the same performance level as adopting *weakestLink*.

[3]The benchmark is running on a AWS EC2 instance (t3a.large), the implementation is writing with Haskell and using criterion library for evaluation. Implementation details could refer to https://bitbucket.org/wuhao29/informative-partial-argument/src/master/.

[4]$td = 20$, $ad = 100$, $lid \in \{10, \ldots, 500\}$

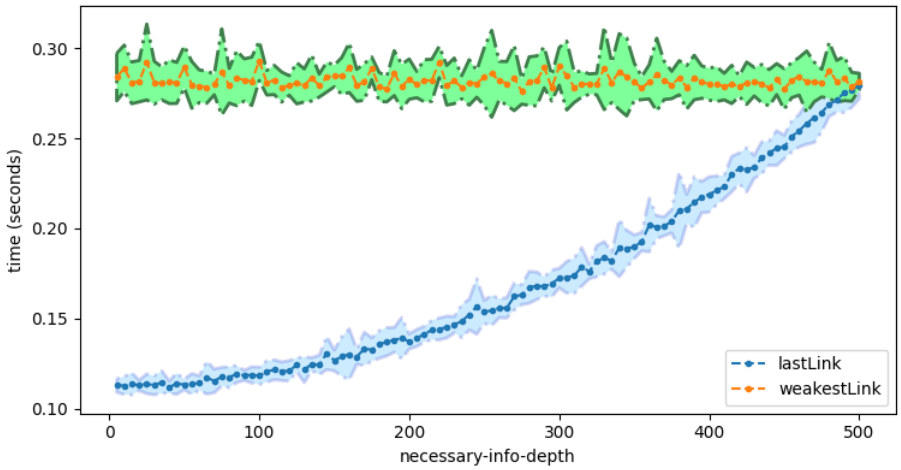

Figure 2: The performances as LastLink-Info-Depth increasing.

- Benchmark Overhead: Set td and lid to be control variables and increase the value of ad at a given step[5]. The benchmark includes 100 defeasible theories. As shown in Figure 3,

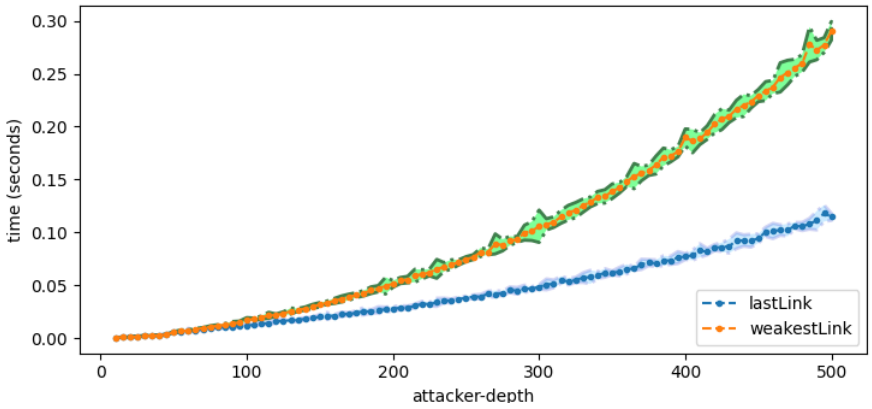

Figure 3: The performance as Attacker-Depth is increasing.

the depth of the first defeasible rule in all attackers is a constant. The x-axis is the length of all attackers, while the y-axis is the time required to complete the computation. As the value of attacker depth increases, the performance of using *weakestLink* is increasing considerably compared to using *lastLink*. When the attacker depth is small enough, it is possible that adopting *lastLink* will take more computational time than adopting *weakLink* due to the overhead brought by extra operation defined by *lastLink*, However, this overhead is negligible as shown in figure 3.

[5] $td = 20$, $ad \in \{5, \ldots, 500\}$, $lid = 5$.

6. Discussion and Future work

The idea of constructing argument by backward chaining can be seen in works such as [5, 6, 7, 8, 9] that discusses dispute derivation for Assumption Based Argumentation (ABA) and the concept of potential argument in [5] is similar to our definition of partial argument. These works focus on evaluating acceptances of claims with respect to different semantics that ABA inherits from abstract argumentation. The argument construction process is embedded in the dialectal procedure defined by corresponding algorithm and cannot be used as a general framework for defining other higher-level definitions.

In this paper, we propose a new framework which factorises the argument construction process and provides modules for improving the expressiveness and the computational efficiency. Based on this framework, a solver could be further developed for tasks such as the ICCMA structured Argumentation track[10], and it is also one of many steps toward solving challenges from real world use cases that involve large dynamic defeasible theories. Future works will mainly focus on the study of the defeasible-theory-level heuristic information so that our framework will be able to optimise the computation at both argument level and the higher-level concept level.

References

[1] P. E. Dunne, M. Wooldridge, Complexity of abstract argumentation, in: Argumentation in artificial intelligence, Springer, 2009, pp. 85–104.

[2] B. Yun, N. Oren, M. Croitoru, Efficient construction of structured argumentation systems, Frontiers in Artificial Intelligence and Applications 326 (2020) 411–418. doi:10.3233/FAIA200525.

[3] M. W. A. Caminada, S. Modgil, N. Oren, Preferences and unrestricted rebut, Computational Models of Argument (2014).

[4] S. Modgil, H. Prakken, The aspic+ framework for structured argumentation: a tutorial, Argument & Computation 5 (2014) 31–62.

[5] R. Craven, F. Toni, Argument graphs and assumption-based argumentation, Artificial Intelligence 233 (2016) 1–59.

[6] P. M. Dung, R. A. Kowalski, F. Toni, Dialectic proof procedures for assumption-based, admissible argumentation, Artificial Intelligence 170 (2006) 114–159.

[7] P. M. Dung, P. Mancarella, F. Toni, Computing ideal sceptical argumentation, Artificial Intelligence 171 (2007) 642–674.

[8] F. Toni, A generalised framework for dispute derivations in assumption-based argumentation, Artificial Intelligence 195 (2013) 1–43.

[9] M. Diller, S. A. Gaggl, P. Gorczyca, Flexible dispute derivations with forward and backward arguments for assumption-based argumentation, in: International Conference on Logic and Argumentation, Springer, 2021, pp. 147–168.

[10] J.-M. Lagniez, E. Lonca, J.-G. Mailly, J. Rossit, The fourth international competition on computational models of argumentation (2020).