Estimating Stability for Efficient Argument-based Inquiry - Proofs

Daphne Odekerken[1,2], AnneMarie Borg[1], and Floris Bex[1,3]

[1]Department of Information and Computing Sciences, Utrecht University
[2]National Police Lab AI, Netherlands Police
[3]Tilburg Institute for Law, Technology and Society, Tilburg University

This document gives the proofs of the propositions stated in [2].

1 Preliminaries

In this section, we introduce the base argumentation framework for which we study the stability problem. We use a variation on ASPIC$^+$ [3], a framework for structured argumentation that evaluates arguments with Dung-style semantics [1]. From ASPIC$^+$, we use the concepts of a logical language, a knowledge base and a set of defeasible rules. We add the notions of topic and queryable literals. The *topic* is the literal for which we want to know the stability status. *Queryables* are those literals in the logical language which can be obtained by querying the client agent. We add the notion of queryables since they restrict the possibilities of updating the knowledge base.

Definition 1 (Argumentation Setup)**.** An **argumentation setup** AS is a tuple $AS = (\mathcal{L}, \mathcal{R}, \mathcal{Q}, \mathcal{K})$ where:

- $\mathcal{L}$ is a finite propositional language, closed under classical negation ($\neg$). The literalswill be denoted by lower-case letters. We write $a = -b$ iff $a = \neg b$ or $b = \neg a$.
- $\mathcal{R}$ is a finite set of defeasible rules $a_1, \ldots, a_m \Rightarrow c$ such that $\{a_1, \ldots, a_m, c\} \subseteq \mathcal{L}$. Where $r \in \mathcal{R}$, $\texttt{ants}(r) = \{a_1, \ldots, a_m\}$ are the antecedents of r and $\texttt{cons}(r) = c$ is its consequent. We refer to a rule with consequent c as "a rule for c".
- $\mathcal{Q} \subseteq \mathcal{L}$ is a set of queryable literals, s.t. $l \in \mathcal{Q}$ iff $-l \in \mathcal{Q}$.
- $\mathcal{K} \subseteq \mathcal{Q}$ is the knowledge base, which is required to be consistent: if $l \in \mathcal{K}$ then $-l \notin \mathcal{K}$.

Based on an argumentation setup, arguments can be constructed from $\mathcal{R}$ and $\mathcal{K}$, as formally defined in Definition 2.

Definition 2 (Argument)**.** Let $AS = (\mathcal{L}, \mathcal{R}, \mathcal{Q}, \mathcal{K})$ be an argumentation setup. We denote by $Arg(AS)$ the set of **arguments** that can be inferred from AS. We distinguish two types of arguments. An argument A inferred from AS is:

- an **observation-based argument** c iff $c \in \mathcal{K}$.
  The set of premises $\texttt{prem}(A)$ of A is $\{c\}$.
  The conclusion $\texttt{conc}(A)$ of A is c.
  The set of subarguments $\texttt{sub}(A)$ of A is $\{c\}$.
  The height $\texttt{h}(A)$ of A is 0.
  The set of defeasible rules $\texttt{def-rules}(A)$ of A is $\emptyset$.
  The set of direct subarguments $\texttt{dirsub}(A)$ of A is $\emptyset$.

- a **rule-based argument** $A_1,\ldots,A_m \Rightarrow c$ iff for each $i \in [1\,..\,m]$: A_i is in $Arg(AS)$ and has conclusion c_i and there is a rule $r: c_1,\ldots,c_m \Rightarrow c$ in $\mathcal{R}$.
  The set of premises $\texttt{prem}(A)$ of A is $\texttt{prem}(A_1)\cup\ldots\cup\texttt{prem}(A_m)$.
  The conclusion $\texttt{conc}(A)$ of A is c.
  The set of subarguments $\texttt{sub}(A)$ of A is $\texttt{sub}(A_1)\cup\ldots\cup\texttt{sub}(A_m)\cup\{A\}$.
  The height $\texttt{h}(A)$ of A is $1+\max(\texttt{h}(A_1),\ldots,\texttt{h}(A_m))$.
  The set of defeasible rules $\texttt{def-rules}(A)$ of A is $\texttt{def-rules}(A_1)\cup\ldots\cup\texttt{def-rules}(A_m)\cup\{r\}$.
  The top rule $\texttt{top-rule}(A)$ is r.
  The set of direct subarguments $\texttt{dirsub}(A)$ of A is $\{A_1,\ldots,A_m\}$.

We refer to an argument with conclusion c as "an argument for c". We refer to a rule-based argument with top rule r as "an argument based on r".

Note that we do not require the arguments to be non-circular, therefore, there may be an argument A for some literal $l \in \mathcal{L}$ such that A has a proper subargument $A' \in \texttt{sub}(A) - \{A\}$ with $\texttt{conc}(A') = \texttt{conc}(A) = l$. We formally define circularity in Definition 3.

Definition 3 (Circularity)**.** Let $\langle A_1,\ldots,A_m\rangle$ be a sequence of arguments. $\langle A_1,\ldots,A_m\rangle$ has the **circularity property** iff $A_1 \neq A_m$, $\texttt{conc}(A_1) = \texttt{conc}(A_m)$ and for each $i \in [1\,..\,m-1]$: $A_i \in \texttt{dirsub}(A_{i+1})$. Let A be an argument. A is **circular** iff there is a subargument $A' \in \texttt{sub}(A)$ such that there is a sequence $\langle A_1,\ldots,A'\rangle$ with the circularity property. A is non-circular iff A is not circular.

Example 1. [Circular arguments] Let $AS = (\mathcal{L},\mathcal{R},\mathcal{Q},\mathcal{K})$ be an argumentation setup with $\mathcal{L} = \{a,\neg a,b,\neg b\}$, $\mathcal{R} = \{a \Rightarrow b, b \Rightarrow b\}$, $\mathcal{Q} = \{a,\neg a\}$ and $\mathcal{K} = \{a\}$. Then the arguments a and $a \Rightarrow b$ in $Arg(AS)$ are not circular. $Arg(AS)$ contains an infinite number of circular arguments as well, for example $[a \Rightarrow b] \Rightarrow b$ and $[[[[[a \Rightarrow b] \Rightarrow b] \Rightarrow b] \Rightarrow b] \Rightarrow b] \Rightarrow b$.

As illustrated in Example 1, for a given argumentation setup $AS = (\mathcal{L},\mathcal{R},\mathcal{Q},\mathcal{K})$, there may be circular arguments in $Arg(AS)$, which are created by infinitely applying some rule $r \in \mathcal{R}$. As a result, these arguments have infinite height. However, as we show in Lemma 1, each circular argument can be turned into a non-circular argument by removing the circular part. In Lemma 2, we prove that all arguments that are non-circular have finite height. These properties will be useful in many proofs based on induction later in this paper.

Lemma 1 (Existence of non-circular arguments)**.** *Let $AS = (\mathcal{L},\mathcal{R},\mathcal{Q},\mathcal{K})$ be an argumentation setup and literal $l \in \mathcal{L}$. There is an argument for l in $Arg(AS)$ iff there is a non-circular argument for l in $Arg(AS)$.*

Proof. We start with the proof from left to right: assume that there is an argument A for l in $Arg(AS)$. If A is non-circular, then there trivially is an non-circular argument for l in $Arg(AS)$. Alternatively, assume that A is circular by Definition 3. Then by Definition 3, there is a sequence $\langle A_1,\ldots,A_m\rangle$ such that $A_m \in \texttt{sub}(A)$, $\texttt{conc}(A_1) = \texttt{conc}(A_m)$, $A_1 \neq A_m$ and for each $i \in [1\,..\,m-1]$: $A_i \in \texttt{sub}(A_{i+1})$ (circularity property). Let $\langle A_1,\ldots,A_m\rangle$ be an arbitrary sequence with the circularity property. Let A' be the argument obtained by replacing A_m by A_1 in A. $A_1 \in \texttt{sub}(A_m) \subseteq Arg(AS)$, so A' is an argument in $Arg(AS)$. Furthermore, $\texttt{conc}(A_1) = \texttt{conc}(A_m)$, so A_1 is an argument for $\texttt{conc}(A_m)$, so A' is an argument for l. Since $A_m \notin \texttt{sub}(A')$, we know that the sequence $\langle A_1,\ldots,A_m\rangle$ cannot cause A' to satisfy the circularity property (in contrast to our original argument A). We chose $\langle A_1,\ldots,A_m\rangle$ arbitrary, so we can repeatedly apply this reasoning for all sequences with the circularity property until there is no such sequence any more and the resulting argument for l is non-circular.

The proof from right to left is trivial: each non-circular argument is an argument, so if there is a non-circular argument for l in $Arg(AS)$, then there is an argument for l in $Arg(AS)$. □

Lemma 2 (Finite height and non-circularity)**.** *Given an argument A, if A is non-circular, then $\texttt{h}(A) \neq \infty$.*

Proof. Suppose that A is non-circular. This implies that there is no sequence $\langle A_1,\ldots,A_m\rangle$ such that $A_m \in \texttt{sub}(A)$, $\texttt{conc}(A_1) = \texttt{conc}(A_m)$, $A_1 \neq A_m$ and for each $i \in [1\,..\,m-1]$: $A_i \in \texttt{dirsub}(A_{i+1})$. Let $\langle A_1,\ldots,A\rangle$ be the longest possible sequence such that for each $i \in [1\,..\,m-1]$: $A_i \in \texttt{dirsub}(A_{i+1})$

First, we show by contradiction that for each $r \in \mathcal{R}$, there is at most one $A_i \in \langle A_1,\ldots,A\rangle$ such that $\texttt{top-rule}(A_i) = r$. Suppose that there are some A_i, A_j such that $A_i, A_j \in \langle A_1,\ldots,A\rangle$, $\texttt{top-rule}(A_i) =$

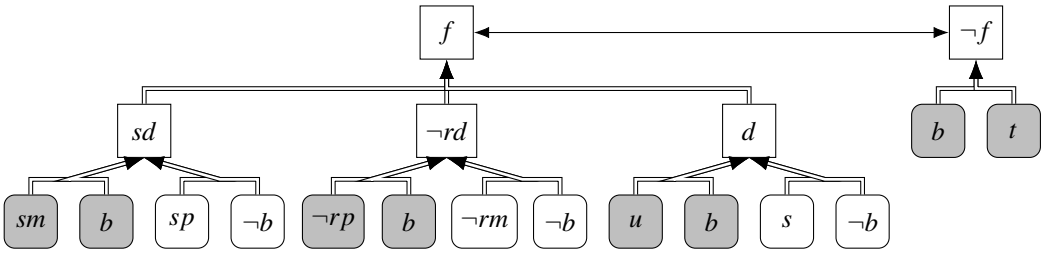

Figure 1: Example of an argumentation setup AS from the law enforcement domain. Note that the literals b and $\neg b$ are visualised multiple times and attacks between them are omitted for clarity.

$\texttt{top-rule}(A_j)$ and $A_i \neq A_j$. Without loss of generality, we assume that $i < j$. Now consider the sequence $\langle A_i, \ldots, A_j \rangle$ (which is a subsequence of $\langle A_1, \ldots, A \rangle$). We have that $\texttt{conc}(A_i) = \texttt{conc}(A_j)$, $A_i \neq A_j$ and for each $k \in [i \,..\, j-1] : A_k \in \texttt{dirsub}(A_{k+1})$; by Definition 3, the sequence $\langle A_i, \ldots, A_j \rangle$ satisfies the circularity property. Since $A_j \in \texttt{sub}(A)$, this implies that A is circular, which contradicts our initial assumption. To conclude, for each $r \in \mathcal{R}$, there is at most one $A_i \in \langle A_1, \ldots, A_m \rangle$ having $\texttt{top-rule}(A_i) = r$.

Then by definition of height, $\texttt{h}(A) \leq |\mathcal{R}|$. By Definition 1, $\mathcal{R}$ is a finite set, so for each non-circular argument A: $\texttt{h}(A) \neq \infty$. □

Arguments can attack each other, as formally defined in Definition 4. Our definition of attack corresponds to rebuttal in ASPIC$^+$ [3].

Definition 4 (Attack)**.** Let $AS = (\mathcal{L}, \mathcal{R}, \mathcal{Q}, \mathcal{K})$ be an argumentation setup. For two arguments $A, B \in Arg(AS)$ we say that A **attacks** B iff A's conclusion is c and either:

- **attack on conclusion**: $-c$ is the conclusion of B and $-c \notin \mathcal{K}$.
- **attack on subargument**: $-c$ is the conclusion of a subargument B' of B such that $B' \neq B$ and $-c \notin \mathcal{K}$.

We write that "A attacks B on B'" if A attacks B, $B' \in \texttt{sub}(B)$ and $\texttt{conc}(A) = -\texttt{conc}(B')$.

From Definition 4 it directly follows that observation-based arguments cannot be attacked.

Example 2 (Online trade fraud, following [4])**.** Let $AS = (\mathcal{L}, \mathcal{R}, \mathcal{Q}, \mathcal{K})$, visualised in Figure 1, be an argumentation setup in the domain of online trade fraud, which concerns cases such as fake web shops and malicious second-hand traders on platforms such as eBay. $\mathcal{L}$ consists of the following literals and their negations: b: complainant (i.e. the client agent) tried to buy a product (as opposed to selling a product); sm: complainant sent money; sp: complainant sent product; rp: complainant received product; rm: complainant received money; u: suspicious url; s: screenshot of payment; t: trusted web shop; sd: complainant delivered; rd: complainant received delivery; d: deception; f: fraud. Squares represent literals from $\mathcal{L}$, rounded squares are queryable literals (from $\mathcal{Q}$) and literals in $\mathcal{K}$ are shaded. Rules are represented by double-lined arrows and attacks as single-lined arrows. $Arg(AS)$ includes an argument for f based on the rule $sd, \neg rd, d \Rightarrow f$ and an argument for $\neg f$ based on $b, t \Rightarrow \neg f$. These arguments attack each other.

We base the evaluation of arguments on the grounded semantics from [1]. We chose grounded semantics since it is characterised by a single extension and its sceptical nature fits with the conservativity of the processes we model.

Definition 5 (Grounded Extension)**.** Let $AS = (\mathcal{L}, \mathcal{R}, \mathcal{Q}, \mathcal{K})$ be an argumentation setup and let $S \subseteq Arg(AS)$. S is said to be **conflict-free** iff there are no $A, B \in S$ such that A attacks B. S **defends** $A \in Arg(AS)$ iff for each $B \in Arg(AS)$ that attacks A there is a $C \in S$ that attacks B. S is **admissible** iff it is conflict-free and defends all its arguments. S is a **complete extension** iff it is admissible and contains all the arguments it defends. The **grounded extension** $G(AS)$ is the least (w.r.t. $\subseteq$) complete extension.

Next, we use the notion of grounded extension to define the acceptability of a literal in an argumentation setup.

Definition 6 (Acceptability)**.** Let $AS = (\mathcal{L}, \mathcal{R}, \mathcal{Q}, \mathcal{K})$ be an argumentation setup. The acceptability of literal $l \in \mathcal{L}$ given AS is:

- **unsatisfiable** iff there is no argument for l in $Arg(AS)$;
- **defended** iff there exists an argument for l in $Arg(AS)$, which is also in the grounded extension $G(AS)$;
- **out** iff there exists an argument for l in $Arg(AS)$, but each argument for l in $Arg(AS)$ is attacked by an argument in the grounded extension $G(AS)$;
- **blocked** iff there exists an argument for l in $Arg(AS)$, but no argument for l is in the grounded extension $G(AS)$ and at least one argument for l is not attacked by an argument in $G(AS)$.

These acceptability statuses are complementary: e.g. if a literal is not unsatisfiable, defended or out, then it is blocked.

Example 3 (Example 2 continued)**.** In the argumentation setup AS from Figure 1, $G(AS)$ contains (unattacked) arguments for sm, b, $\neg rp$, u, t, sd, $\neg rd$ and d, so these literals are defended in AS. There are arguments for f and $\neg f$ in $Arg(AS)$ that attack each other, but these are not attacked or defended by any argument in $G(AS)$, so f and $\neg f$ are blocked in AS. All other literals in $\mathcal{L}$ are unsatisfiable in AS, since there is no argument for these literals in $Arg(AS)$.

In Lemmas 3 and 4, we give a more precise specification of which arguments are either in the grounded extension (Lemma 3) or attacked by an argument in the grounded extension (Lemma 4). These specifications will be useful for many proofs in this paper.

Lemma 3 (Specification "in" grounded extension)**.** *Given an argumentation setup $AS = (\mathcal{L}, \mathcal{R}, \mathcal{Q}, \mathcal{K})$, an argument $A \in Arg(AS)$ is in the grounded extension $G(AS)$ iff each argument attacking A is attacked by an observation-based argument.*

Proof. The proof from right to left is trivial: observation-based arguments cannot be attacked (Definition 4), so each observation-based argument is in $G(AS)$. If each argument attacking A is attacked by an observation-based argument, then A is defended by $G(AS)$, so $A \in G(AS)$.

We now prove the left-to-right part by contradiction. Assume that $A \in G(AS)$ and that there is an argument B attacking A, and B is not attacked by an observation-based argument. This implies that A is rule-based: if A would be observation-based, then there would be no argument in $Arg(AS)$ attacking A. By Definition 5, $G(AS) \subseteq Arg(AS)$ is a set of arguments that is complete, i.e. $G(AS)$ is conflict-free, each argument from $G(AS)$ is defended by $G(AS)$ and each argument that is defended by $G(AS)$ is in $G(AS)$.

Now let $T = \{C \in G(AS) \mid \text{ there is a } C' \in \texttt{sub}(C) \text{ s.t. } C' \text{ attacks } B\}$.

We will show that each argument in T is rule-based: an argument C is only added to T if it has a subargument C' that attacks B. We assumed that B is not attacked by an observation-based argument, which implies that C' must be rule-based. If C would be observation-based, then it would not have a rule-based subargument C', so C must be rule-based as well.

Also note that $A \in T$, as we show next: A is attacked by B, so there is a subargument $A' \in \texttt{sub}(A)$ such that $\texttt{conc}(A') = -\texttt{conc}(B)$ and $\texttt{conc}(A') \notin \mathcal{K}$. If B would be observation-based, then $B \in G(AS)$, which would contradict the assumption that $A \in G(AS)$ (since $G(AS)$ is conflict-free). This implies that B is rule-based. Given that B is rule-based, we have that A' also attacks B on its conclusion. $\texttt{conc}(A') \notin \mathcal{K}$, so A' is rule-based. Given that $A' \in \texttt{sub}(A)$, A' defends A and A' is rule-based, we have that $A \in T$.

Next, we consider the set $S = G(AS) - T$. Note that $A \notin S$ since $A \in T$. Then we know that T is not empty, so in combination with the facts that $S \subseteq G(AS)$ and $S = G(AS) - T$ we derive that $S \subset G(AS)$. Next, we prove that S is complete, that is: S is conflict-free, each argument from S is defended by S and each argument that is defended by S is in S.

- S is conflict-free: $S \subset G(AS)$ and $G(AS)$ is conflict-free.

- Now we prove that each argument from S is defended by S. Suppose that there exists an argument $D \in S$ such that D is not defended by S: there is an argument E attacking D and each argument F attacking E is not in S. E attacks D, so there is an argument $D' \in \texttt{sub}(D)$ such that $\texttt{conc}(D') = -\texttt{conc}(E)$. Since $D \in G(AS)$, we know that $G(AS)$ defends D, so given that E attacks D, there must be an argument in $Arg(AS)$ that attacks E. So $\texttt{conc}(E) \notin \mathcal{K}$. But then D' attacks E, so D' (which possibly equals D) defends D. We assumed that D is not defended by S, so $D' \notin S$. $D \in G(AS)$ and $G(AS)$ is complete, so D is defended by $G(AS)$. Then each argument defending D must be in $G(AS) - S = T$. This implies that $D' \in T$, so by definition of T, D' has a subargument that defends A. But since D' is a subargument of D, this implies that D has a subargument that defends A, so $D \in T$. This contradicts our assumption that $D \in S$, so each argument from S is defended by S.

- Finally, we prove that each argument that is defended by S is in S. Suppose that there exists an argument $D \in Arg(AS)$ such that S defends D and $D \notin S$. If S defends D, then $G(AS)$ defends D (since $S \subset G(AS)$); therefore $D \in G(AS)$. $D \notin S$, so D must be in T. Then by definition of T, there is a subargument $D' \in \texttt{sub}(D)$ such that D' attacks B. Let $B' \in \texttt{sub}(B)$ be the subargument of B on which D' attacks B': $\texttt{conc}(B') = -\texttt{conc}(D')$. Remember that each argument in T is rule-based. $D' \in T$, therefore $\texttt{conc}(D') \notin \mathcal{K}$, so B' attacks D' as well. $D' \in \texttt{sub}(D)$ and $\texttt{conc}(B') = -\texttt{conc}(D')$, so B' attacks D. S defends D, so there must be an argument E in S attacking B'. But then E would attack B as well, since $B' \in \texttt{sub}(B)$. Then by definition of T, $E \in T$, which contradicts our assumption that $E \in S$. As a result, each argument that is defended by S is in S.

To conclude, there is a set $S \subset G(AS)$ such that S is complete. This contradicts our assumption that $G(AS)$ is the grounded extension: The grounded extension should be minimal w.r.t. set inclusion (Definition 5). So if $A \in G(AS)$, then each argument attacking A is attacked by an observation-based argument. □

Lemma 4 (Specification "out" arguments)**.** *Given an argumentation setup $AS = (\mathcal{L}, \mathcal{R}, \mathcal{Q}, \mathcal{K})$, an argument $A \in Arg(AS)$ is attacked by an argument in the grounded extension $G(AS)$ (A is "out"), iff A is attacked by an observation-based argument.*

Proof. We first prove this from right to left: if an argument $A \in Arg(AS)$ is attacked by an observation-based argument B, then $B \in G(AS)$, since B cannot be attacked. So A is attacked by an argument in the grounded extension.

Now we prove the left-to-right part by contradiction. Suppose that an argument $A \in Arg(AS)$ is attacked by an argument B in the grounded extension, but A is not attacked by an observation-based argument. Then there is a subargument $A' \in \texttt{sub}(A)$ such that $\texttt{conc}(A') = -\texttt{conc}(B)$, $\texttt{conc}(A') \notin \mathcal{K}$ and $\texttt{conc}(B) \notin \mathcal{K}$. B attacks A on A', but A' attacks B as well. B is in $G(AS)$, so each argument attacking B is attacked by an observation-based argument (Lemma 3). So A' must be attacked by an observation-based argument: there is a subargument $A'' \in \texttt{sub}(A')$ such that $-\texttt{conc}(A'') \in \mathcal{K}$. But if $A'' \in \texttt{sub}(A')$ and $A' \in \texttt{sub}(A)$ then $A'' \in \texttt{sub}(A)$, which implies that A is attacked by an observation-based argument as well. This contradicts our assumption that A is not attacked by an observation-based argument.

To conclude, an argument $A \in Arg(AS)$ is attacked by an argument in the grounded extension $G(AS)$, iff A is attacked by an observation-based argument. □

2 Stability

Using the notion of acceptability (Definition 6), we can determine whether a literal $l \in \mathcal{L}$ can be accepted in a given argumentation setup $AS = (\mathcal{L}, \mathcal{R}, \mathcal{Q}, \mathcal{K})$. However, by adding more information, l's acceptability status may change. Informally, l is *stable* in AS if its acceptability status cannot change by adding any combination of queryables to the knowledge base - provided that the resulting knowledge base is consistent. Next, we define future setups, which specify how information can be added to AS.

Definition 7 (Future setups)**.** The set of **future setups** $F(AS)$ of an argumentation setup $AS = (\mathcal{L}, \mathcal{R}, \mathcal{Q}, \mathcal{K})$ contains all argumentation setups $AS' = (\mathcal{L}, \mathcal{R}, \mathcal{Q}, \mathcal{K}')$ with $\mathcal{K} \subseteq \mathcal{K}'$.

If an argument A can be inferred from a given argumentation setup $AS = (\mathcal{L}, \mathcal{R}, \mathcal{Q}, \mathcal{K})$, then A can be inferred from each future argumentation setup $AS' \in F(AS)$. We formally prove this in Lemma 5.

Lemma 5 (Argument persistence in future setups)**.** *Let $AS = (\mathcal{L}, \mathcal{R}, \mathcal{Q}, \mathcal{K})$ be an argumentation setup and let A be an argument in $Arg(AS)$. Then for each $AS' \in F(AS)$: $A \in Arg(AS')$.*

Proof. Suppose that $AS = (\mathcal{L}, \mathcal{R}, \mathcal{Q}, \mathcal{K})$ is an argumentation setup and $A \in Arg(AS)$. Let $AS' = (\mathcal{L}, \mathcal{R}, \mathcal{Q}, \mathcal{K}') \in F(AS)$ be an arbitrary future setup. By definition of future setups (Definition 7), $\mathcal{K} \subseteq \mathcal{K}'$. From the definition of arguments (Definition 2), it follows that arguments are constructed only based on their knowledge base and rule set, so given that A can be constructed from $\mathcal{K}$ and $\mathcal{R}$, A can also be constructed from $\mathcal{K}'$ and $\mathcal{R}$. This implies that $A \in Arg(AS')$. □

Note that argumentation setup AS always belongs to the set of future setups $F(AS)$. Further recall from Definition 1 that $\mathcal{K}'$ must be consistent since AS' is an argumentation setup. Using the notion of future setups, we now define stability.

Definition 8 (Stability)**.** Let $AS = (\mathcal{L}, \mathcal{R}, \mathcal{Q}, \mathcal{K})$ be an argumentation setup. A literal $l \in \mathcal{L}$ is **stable** in AS iff any of the following holds:

- for each $AS' \in F(AS)$, l is **unsatisfiable** in AS'; or
- for each $AS' \in F(AS)$, l is **defended** in AS'; or
- for each $AS' \in F(AS)$, l is **out** in AS'; or
- for each $AS' \in F(AS)$, l is **blocked** in AS'.

Example 4 (Example 3 continued)**.** In our running example, the topic f is stable. By querying the client agent, we could obtain more information; $F(AS)$ for example contains an argumentation setup with knowledge base $\mathcal{K}' = \mathcal{K} \cup \{\neg sp\} = \{sm, b, \neg rp, u, t, \neg sp\}$. However, adding information does not influence f's acceptability status: for each AS' in $F(AS)$, f is blocked in AS'. Therefore, f is stable in AS.

Determining stability is CoNP-hard. This can be shown by a polynomial-time reduction from the CoNP-complete problem UNSAT.

Proposition 1. *[Complexity of stability problem] Given an argumentation setup $AS = (\mathcal{L}, \mathcal{R}, \mathcal{Q}, \mathcal{K})$, the problem of deciding if a literal $l \in \mathcal{L}$ is stable in AS is CoNP-hard.*

Proof. Consider an argumentation setup $AS = (\mathcal{L}, \mathcal{R}, \mathcal{Q}, \mathcal{K})$ and let STABILITY be the problem of deciding if a literal $l \in \mathcal{L}$ is stable in AS. We will show that STABILITY is CoNP-hard by giving a polynomial-time computable reduction f from the known CoNP-hard problem UNSAT.

Consider a boolean expression $\phi = (c_{11} \vee \ldots \vee c_{1k}) \wedge \ldots \wedge (c_{n1} \vee \ldots \vee c_{nm})$ in conjunctive normal form (CNF); let $\mathcal{C}$ be the set containing all literals in ϕ and let v_I be the valuation function for propositional classical logic given the truth assignment function $I : \mathcal{C} \to \{\text{True}, \text{False}\}$. In the UNSAT problem, the goal is to decide, given a boolean expression $\phi = (c_{11} \vee \ldots \vee c_{1k}) \wedge \ldots \wedge (c_{n1} \vee \ldots \vee c_{nm})$ in CNF, if $v_I(\phi)$ is False for every truth assignment I.

First, we will give a reduction function from UNSAT to the STABILITY problem. Let f be the function that converts an arbitrary formula $\phi = (c_{11} \vee \ldots \vee c_{1k}) \wedge \ldots \wedge (c_{n1} \vee \ldots \vee c_{nm})$ in CNF to an argumentation setup $AS = (\mathcal{L}, \mathcal{R}, \mathcal{Q}, \mathcal{K})$ (illustrated in Figure 2) such that:

- $\mathcal{L}$ consists of:
  - (1) For each literal in $\mathcal{C}$: the literal and its negation; and
  - (2) For each clause in ϕ: a clause-specific literal and its negation; and
  - (3) A topic literal and its negation.

  Formally: $\mathcal{L} = \mathcal{C} \cup \{-c_{ij} \mid c_{ij} \in \mathcal{C}\}$ (1) $\cup \{l_i \mid \exists j : c_{ij} \in \mathcal{C}\} \cup \{-l_i \mid \exists j : c_{ij} \in \mathcal{C}\}$ (2) $\cup \{t, \neg t\}$ (3);
- $\mathcal{R}$ consists of:
  - (1) For each clause i and for each literal c_{ij} in this clause: $c_{ij} \Rightarrow l_i$; and

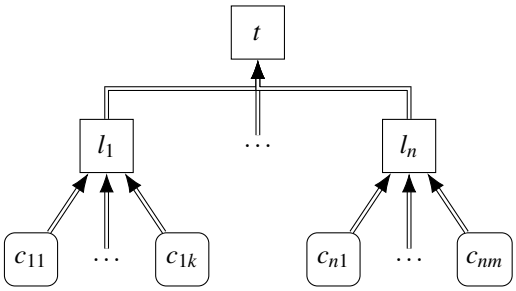

Figure 2: Reduction UNSAT.

(2) $(l_1, \ldots, l_n) \Rightarrow t$ for t.

Formally: $\mathcal{R} = \{c_{ij} \Rightarrow l_i \mid c_{ij} \in \mathcal{C}\}$ (1) $\cup \{(l_1, \ldots, l_n) \Rightarrow t\}$ (2).

- $\mathcal{Q}$ contains all literals occurring in the CNF ϕ and their negations: $\mathcal{Q} = \{c_{ij} \mid c_{ij} \in \mathcal{C} \vee -c_{ij} \in \mathcal{C}\}$; and
- $\mathcal{K}$ is empty: $\mathcal{K} = \emptyset$.

We now examine the time needed for computing the reduction $f(\phi)$ for an arbitrary formula ϕ in CNF. For each literal c_{ij} occurring in ϕ, two literals are added to the language $\mathcal{L}$, one rule is added to $\mathcal{R}$ and two literals are added to $\mathcal{Q}$. Furthermore, for each clause two literals are added to $\mathcal{L}$. Finally, the topic literal and its negation are added to $\mathcal{L}$ and a single rule for the topic literal is added to $\mathcal{R}$. Since the number of clauses cannot exceed the number of literals occurring in ϕ, this reduction can be computed $\mathcal{O}(|\mathcal{C}|)$ operations; hence the reduction f can be computed in polynomial time.

Next, we prove that there is no truth assignment I such that $v_I(\phi) = \text{True}$ iff t is stable in AS.

We start by proving this from left to right, by contraposition. Suppose that there exists a truth assignment I for $\phi = (c_{11} \vee \ldots \vee c_{1k}) \wedge \ldots \wedge (c_{n1} \vee \ldots \vee c_{nm})$ such that $v_I(\phi) = \text{True}$. We will show that t cannot be stable. The knowledge base $\mathcal{K}$ is empty, so by Definition 2, $Arg(AS) = \emptyset$. Then there is no argument for t in $Arg(AS)$. Now let $\mathcal{K}' = \{c_{ij} \in \mathcal{K} \mid v_I(c_{ij}) = \text{True}\}$ and let AS' be $\{\mathcal{L}, \mathcal{R}, \mathcal{Q}, \mathcal{K}'\}$. $v_I(\phi) = \text{True}$, hence for each $i \in [1 \,..\, n]$, $v_I(c_{i1} \vee \ldots \vee c_{ik}) = \text{True}$. Consider an arbitrary $i \in [1 \,..\, n]$. $v_I(c_{i1} \vee \ldots \vee c_{ik}) = \text{True}$, so there is a $j \in [1 \,..\, k]$ such that $v_I(c_{ij}) = \text{True}$. Then by definition of $\mathcal{K}'$, $c_{ij} \in \mathcal{K}'$. By Definition 2, there is an observation-based argument for c_{ij}. Furthermore, there is a rule $c_{ij} \Rightarrow l_i$ in $\mathcal{R}$; therefore by Definition 2, there is a rule-based argument for l_i in $Arg(AS')$. Since we chose i arbitrary, for each $i \in [1 \,..\, n]$, there is a rule-based argument for l_i in $Arg(AS')$. Then by Definition 2, there is an argument for t based on the rule $(l_1, \ldots, l_n) \Rightarrow t$ in $Arg(AS')$.

There is no argument for t in $Arg(AS)$ and $AS \in F(AS)$, so by Definition 6, t is unsatisfiable in AS. There is an argument for t in $Arg(AS')$; this implies that t is not unsatisfiable in AS'. Then by Definition 8, t is not stable in AS.

Now, we prove by contraposition that t is stable in AS if there is no truth assignment I such that $v_I(\phi) = \text{True}$.

Suppose that t is not stable in AS. Then by Definition 8, t is not unsatisfiable in every $AS' \in F(AS)$. However, t is unsatisfiable in AS, since there is no argument (for t) in $Arg(AS)$: $\mathcal{K} = \emptyset$. This implies that there exists an $AS' = (\mathcal{L}, \mathcal{R}, \mathcal{Q}, \mathcal{K}')$ in $F(AS)$ such that there is an argument for t in $Arg(AS')$.

$t \notin \mathcal{Q}$, so the argument for t must be rule-based. The only rule for t in $\mathcal{R}$ is $(l_1, \ldots, l_n) \Rightarrow t$, so there is an argument for t based on $(l_1, \ldots, l_n) \Rightarrow t$ in $Arg(AS')$. By Definition 2, for each $i \in [1 \,..\, n]$, there is an argument for l_i in $Arg(AS')$.

Consider an arbitrary $i \in [1 \,..\, n]$. $l_i \notin \mathcal{Q}$, so there must be a rule-based argument for l_i. Let $\{r_{i1}, \ldots, r_{ik}\}$ be the rules for l_i in $\mathcal{R}$. Let $r_{ij} : c_{ij} \Rightarrow l_i$ (with $i \in [1 \,..\, k]$) be an arbitrary rule such that there is an argument based on r_{ij} in $Arg(AS')$. Then there must be an argument for c_{ij} in $Arg(AS')$. This argument must be observation-based, hence $c_{ij} \in \mathcal{K}'$.

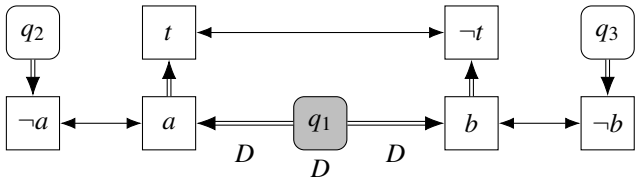

Figure 3: D/B is irrelevant in [5] *Case B lit. A*.

Let I be a truth assignment such that:

$$I(c_{ij}) = \begin{cases} \text{True} & \text{if } c_{ij} \in \mathcal{K}' \\ \text{False} & \text{if } c_{ij} \notin \mathcal{K}'. \end{cases}$$

Given that $c_{ij} \in \mathcal{K}'$, we have that $v_I(c_{ij}) = \text{True}$ and therefore $v_I(c_{i1} \vee \ldots \vee c_{ik}) = \text{True}$ as well. Since we chose i arbitrary, this is the case for every i in $[1\,..\,n]$. Then $v_I((c_{11} \vee \ldots \vee c_{1k}) \wedge \ldots \wedge (c_{n1} \vee \ldots \vee c_{nm})) = v_I(\phi) = \text{True}$. So there exists a truth assignment I such that $v_I(\phi) = \text{True}$.

We have shown that there exists a reduction function f from UNSAT to the STABILITY problem that can be computed in polynomial time, such that $v_I(\phi)$ is False for every truth assignment I iff $t \in \mathcal{L}$ is stable in $f(\phi) = AS = (\mathcal{L}, \mathcal{R}, \mathcal{Q}, \mathcal{K})$. This implies that UNSAT $\leq_p$ STABILITY. UNSAT is CoNP-hard, so STABILITY is CoNP-hard. □

CoNP-hard problems are generally considered intractable (unless P = NP). Given the above results and assuming that P $\neq$ NP, there is no exact polynomial-time algorithm that determines for an arbitrary argumentation setup AS if a literal is stable in AS. This means that an exact algorithm would need exponential time, resulting in infeasible computation time for stability detection of e.g. argumentation setups with large rule sets. Since practical applications require fast computation for arbitrary argumentation setups, we consider a sound polynomial-time approximation algorithm in the next section.

3 Approximating stability

A first approximation algorithm for determining stability in formal argumentation was proposed in [5]. This algorithm assigns a label to literals and rules that it considers to be stable. Each label relates to one of the four cases of stability: U (unsatisfiable); D (defended); O (out); or B (blocked). However, the algorithm is not complete: there exist argumentation setups which are stable but are not labelled as such by the approximation algorithm. Although one example is given in [5], the authors give no precise specification of argumentation setups for which the algorithm does not recognise stability. In the next subsection, we give two additional examples which reveal different issues of the method by [5]. In Sections 3.2 and 3.3, we present a refined algorithm to solve these issues. Subsequently, we will show soundness and conditional completeness and study computational complexity of this refinement in Section 3.4.

3.1 Examples of incompleteness basic algorithm

Figures 3 and 4 illustrate two different issues of the algorithm from [5].

Example 5 (Irrelevant label problem)**.** Figure 3 represents an argumentation setup AS in which q_1, q_2 and q_3 are queryable. q_1 is in the knowledge base and the topic is t. There is an argument for t based on $a \Rightarrow t$ and an argument for $\neg t$ based on $b \Rightarrow \neg t$ in $Arg(AS)$. So for each $AS' \in F(AS)$, t is blocked in AS'. However, t is not recognised as being stable by the algorithm in [5]. The literal q_1 and rules $q_1 \Rightarrow a$ and $q_1 \Rightarrow b$ are correctly labelled D, but the other literals and rules are not labelled by the algorithm. a and b are not labelled because they may become either defended (if $\neg q_2$ resp. $\neg q_3 \in \mathcal{K}'$) or blocked (if q_2 resp. $q_3 \in \mathcal{K}'$). As a result, the rules $a \Rightarrow \neg t$ and $b \Rightarrow t$ are not labelled because they may become either defended or blocked. In all future setups in $F(AS)$, the argument for a is either defended (if $q_2 \notin \mathcal{K}'$) or blocked (if

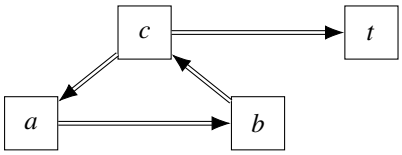

Figure 4: Support Cycle. $\mathcal{Q} = \emptyset$, so $F(AS) = \{AS\}$. $\mathcal{K} = \emptyset$, so there is no argument for t in $Arg(AS)$. However, L does not label t.

$q_2 \in \mathcal{K}'$). Similarly, in every future setup, the argument for b is either defended (if $q_3 \notin \mathcal{K}'$) or blocked (if $q_3 \in \mathcal{K}'$). The algorithm in [5] has a labelling rule *Case B literal A* stating that "$l \in \mathcal{L}$ is labelled B iff $l \in \mathcal{Q}$ and a rule for l and a rule for $-l$ are labelled D or B". However, this rule does not apply: although the rules $a \Rightarrow t$ and $b \Rightarrow \neg t$ will certainly be labelled D or B in a future setup in which we have more information about q_2 and q_3, we do not know the exact label - which is in this case irrelevant.

We will refer to the issue illustrated in Figure 3 as the *irrelevant label problem*. It is caused by the fact that L only assigns a label if there is exactly one possible acceptance status for all future setups, but does not take into account that some acceptability statuses are *impossible* in a future setup.

The next example reveals another issue of the basic algorithm, which we will refer to as the *support cycle problem*.

Example 6 (Support cycle problem)**.** Figure 4 represents an argumentation setup AS in which a, b, c and t are literals that are not queryable and t is the topic literal. None of the literals is queryable, so there is no other future argumentation setup than the current setup (i.e. $F(AS) = \{AS\}$). There is no argument for t in $Arg(AS)$, hence t is unsatisfiable in every future argumentation setup. However, no rule or literal is labelled U since the basic algorithm in [5] only labels a non-queryable literal U if all rules for this literal are labelled U and a rule only gets labelled U if at least one antecedent of that rule is labelled U. Because of this support cycle, there is no place to start labelling.

Due to the irrelevant label problem and the support cycle problem, the approximation algorithm from [5] fails to recognise the stability of some argumentation setups. In an inquiry dialogue context, this would cause an agent to not recognise the termination criterion. As a result, it might ask unnecessary questions. In the next two subsections, we will present a solution to these problems.

3.2 Reasoning with possible future labels

In this section, we present an alternative labelling method, that bypasses the irrelevant label problem by reasoning with possible future labels. Whereas the approximation algorithm presented in [5] relies on a partial labelling function L that assigns at most one label to each literal in $\mathcal{L}$ and rule in $\mathcal{R}$ ($L : \mathcal{L} \cup \mathcal{R} \nrightarrow \{U, D, O, B\}$ where $\nrightarrow$ denotes a partial function), we propose a labelling L' that assigns a quadruple of four booleans $\langle u, d, o, b \rangle$ to each literal and rule. Each boolean corresponds to an acceptability status. Intuitively, the truth value of a boolean belonging to a literal or rule represents the possibility that this literal or rule may become unsatisfiable (u), defended (d), out (o) or blocked (b) in a future argumentation setup.

Similar to the approach in [5], labels of rules depend on the labels of their antecedent literals and labels of literals depend on the labels of rules for that literal. Literals and rules are labelled incrementally, starting from queryable literals and literals for which there is no rule and relabelling literals and rules based on the resulting new labels, until no new label can be added. This procedure is given in Algorithm 1.

Definition 9 (Quadruple labelling L')**.** Let $AS = (\mathcal{L}, \mathcal{R}, \mathcal{Q}, \mathcal{K})$ be an argumentation setup. The **labelling function** $L' : \mathcal{L} \cup \mathcal{R} \to \{0,1\} \times \{0,1\} \times \{0,1\} \times \{0,1\}$ assigns a label $\langle u, d, o, b \rangle$ to each literal or rule in $\mathcal{L} \cup \mathcal{R}$.

Given a literal or rule $x \in \mathcal{L} \cup \mathcal{R}$, we write $\neg u(x)$ [resp. $\neg d(x), \neg o(x), \neg b(x)$] iff the u- [resp. d-, o-, b-] boolean of x's label is False and $u(x)$ [resp. $d(x), o(x), b(x)$] iff the u- [resp. d-, o-, b-] boolean of x's label is True. We say that a rule or literal x is **labelled stable** by L' iff exactly one of the booleans is True: $L'(x)$ is $\langle 1,0,0,0 \rangle$, $\langle 0,1,0,0 \rangle$, $\langle 0,0,1,0 \rangle$ or $\langle 0,0,0,1 \rangle$.

Given a literal $l \in \mathcal{L}$, $L'(l) = \langle u, d, o, b \rangle$ where:

literal cannot become unsatisfiable: $\neg u(l)$ iff:

L-U-a) $l \in \mathcal{K}$; or

L-U-b) there is a rule r for l with $\neg u(r)$.

literal cannot become defended: $\neg d(l)$ iff:

L-D-a) $-l \in \mathcal{K}$; or

L-D-b) $l \notin \mathcal{Q}$ and for each rule r for l: $\neg d(r)$; or

L-D-c) $l \notin \mathcal{Q}$ and there is a rule r' for $-l$ with $\neg u(r')$ and $\neg o(r')$.

literal cannot become out: $\neg o(l)$ iff:

L-O-a) $l \in \mathcal{K}$; or

L-O-b) for each rule r for l: $\neg d(r)$, $\neg o(r)$ and $\neg b(r)$; or

L-O-c) $l \notin \mathcal{Q}$ and for each rule r for l: $\neg o(r)$; or

L-O-d) $l \notin \mathcal{Q}$ and there is a rule r for l with $\neg u(r)$ and $\neg o(r)$.

literal cannot become blocked: $\neg b(l)$ iff:

L-B-a) $l \in \mathcal{Q}$; or

L-B-b) for each rule r for l: $\neg d(r)$ and $\neg b(r)$; or

L-B-c) for each rule r for l: $\neg b(r)$ and for each rule r' for $-l$: $\neg d(r')$ and $\neg b(r')$.

L-B-d) there is a rule r for l with $\neg u(r)$, $\neg o(r)$ and $\neg b(r)$ and for each rule r' for $-l$: $\neg d(r')$ and $\neg b(r')$.

Given a rule $r \in \mathcal{R}$, $L'(r) = \langle u, d, o, b \rangle$ where:

rule cannot become unsatisfiable: $\neg u(r)$ iff:

R-U-a) for each antecedent l of r: $\neg u(l)$.

rule cannot become defended: $\neg d(r)$ iff:

R-D-a) there is an antecedent l of r with $\neg d(l)$.

rule cannot become out: $\neg o(r)$ iff:

R-O-a) for each antecedent l of r: $\neg o(l)$; or

R-O-b) there is an antecedent l of r with $\neg d(l)$ and $\neg o(l)$ and $\neg b(l)$.

rule cannot become blocked: $\neg b(r)$ iff:

R-B-a) for each antecedent l of r: $\neg b(l)$; or

R-B-b) there is an antecedent l of r with $\neg d(l)$ and $\neg b(l)$.

Example 7. We give some intuition on these rules by labelling $AS = (\mathcal{L}, \mathcal{R}, \mathcal{Q}, \mathcal{K})$ illustrated in Figure 5. Some rules apply if (the negation of) a literal is in $\mathcal{K}$ or $\mathcal{Q}$, e.g. q_1 is labelled $\langle 0,1,0,0 \rangle$ by Definition 9 case L-U-a, L-O-b and L-B-a: there is an observation-based argument for q_1 which cannot be attacked in any future setup. The absence of rules for a literal is informative for the acceptability status as well: e.g. l_1 is labelled $\langle 1,0,0,0 \rangle$ by L-D-b, L-O-b/c and L-B-b/c.

Other labels are based on the rules for (the negation of) a literal and propagate properties of (attacks on) subarguments. For example, $q_1 \Rightarrow l_2$ is labelled $\langle 0,1,0,0 \rangle$ by R-U-a, R-O-a and R-B-a and l_2 is labelled $\langle 0,1,0,0 \rangle$ by L-U-b, L-O-c/d and L-B-c/d. Some literals and rules cannot be labelled stable, but still some acceptability status(es) can be excluded: e.g. the rule $q_2 \Rightarrow q_3$ is labelled $\langle 1,1,0,0 \rangle$ by case R-O-a and R-B-a.

Algorithm 1 Labelling procedure

1: **procedure** LABELLING-PROCEDURE($\mathcal{L}, \mathcal{R}, \mathcal{Q}, \mathcal{K}$)
2: Label each literal/rule x in $\mathcal{L}$ and $\mathcal{R}$ as $\langle 1,1,1,1\rangle$
3: TODO-SET = empty set
4: **for** Literal l in $\mathcal{L}$ such that $l \in \mathcal{Q}$ or there is no rule for l in $\mathcal{R}$ **do**
5: Relabel l using Definition 9
6: Add all rules having l as antecedent to TODO-SET
7: **while** TODO-SET is not empty **do**
8: Pop a rule r from TODO-SET
9: Relabel r using Definition 9
10: **if** r's label changed **then**
11: Relabel $\texttt{cons}(r)$ using Definition 9
12: **if** $\texttt{cons}(r)$'s label changed **then**
13: Add all rules having $\texttt{cons}(r)$ as antecedent to TODO-SET
14: Relabel $-\texttt{cons}(r)$ using Definition 9
15: **if** $-\texttt{cons}(r)$'s label changed **then**
16: Add all rules having $-\texttt{cons}(r)$ as antecedent to TODO-SET

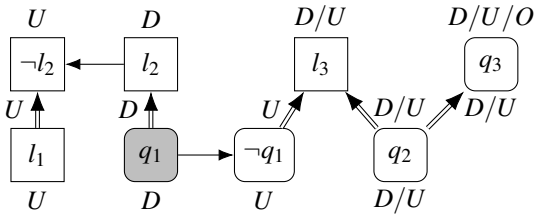

Figure 5: Quadruple labelling example.

Example 8 (Alternative labelling Figure 3)**.** Consider the L' labelling for the argumentation setup from Figure 3. q_1 is in the knowledge base, so by Definition 9, $L'(q_1) = \langle 0,1,0,0\rangle$. Then $L'(q_1 \Rightarrow a) = L'(q_1 \Rightarrow b) = \langle 0,1,0,0\rangle$ by R-U-a, R-O-a and R-B-a. q_2 and q_3 are queryable but not in the knowledge base and there are no rules for q_2 or q_3, so by Case L-O-b and L-B-a: $L'(q_2) = L'(q_3) = \langle 1,1,0,0\rangle$. For the rules $q_2 \Rightarrow \neg b$ and $q_3 \Rightarrow \neg a$, only the d- and u-booleans are True by R-O-a and R-B-a. As a result, for the literals a and b only the d- and b-booleans are True by L-U-b and L-O-c, which implies by R-U-a and R-O-a that $L'(b \Rightarrow t) = L'(a \Rightarrow t) = \langle 0,1,0,1\rangle$. Finally, t is labelled $L'(t) = \langle 0,0,0,1\rangle$ (by L-U-b, L-D-c and L-O-c/d), so t is labelled stable by L'.

In Example 8 we show that t is labelled stable by our labelling function L', while its stability was not detected by the labelling function L from [5]. In general, each literal or rule that is labelled stable by L, is also labelled stable by L', but L' covers more stable setups than L.

Finally, we consider the time complexity of the alternative labelling procedure. As formally shown in Lemma 6, the labelling procedure can be done in polynomial time.

Lemma 6. *The time complexity of Algorithm 1 is* $\mathcal{O}(|\mathcal{L}|^2 \cdot |\mathcal{R}| + |\mathcal{R}|^2)$.

Proof. We will prove this by first showing the amount of time that is required for a *single* execution of a given line (also given in the second column of Table 1). Next, we will consider the number of iterations of each line (third column), multiply them to get the total time required for each line (third column) and combine this into the big-O notation (final row).

In the following, we will denote positive constants by c_i (with $i \in [1\,..\,15]$). Line 2 requires visiting all literals and rules and therefore takes at most $c_1 \cdot (|\mathcal{L}| + |\mathcal{R}|)$ steps. Line 3 takes constant time c_2. Line 4 takes a new literal or rule from $\mathcal{Q}$ or $\mathcal{R}$; a single execution of this line takes constant time c_3. A single execution of line 5 requires labelling a literal, which can be done in $c_4 \cdot |\mathcal{R}|$ time since it requires checking

Line	Time single execution	Max nr. of executions	Total time
2	$c_1 \cdot (\|\mathcal{L}\| + \|\mathcal{R}\|)$	1	$c_1 \cdot (\|\mathcal{L}\| + \|\mathcal{R}\|)$
3	c_2	1	c_2
4	c_3	$\|\mathcal{L}\| + \|\mathcal{R}\|$	$c_3 \cdot (\|\mathcal{L}\| + \|\mathcal{R}\|)$
5	$c_4 \cdot \|\mathcal{R}\|$	$\|\mathcal{L}\| + \|\mathcal{R}\|$	$c_4 \cdot \|\mathcal{R}\| \cdot (\|\mathcal{L}\| + \|\mathcal{R}\|)$
6	$c_5 \cdot \|\mathcal{R}\|$	$\|\mathcal{L}\| + \|\mathcal{R}\|$	$c_5 \cdot \|\mathcal{R}\| \cdot (\|\mathcal{L}\| + \|\mathcal{R}\|)$
7	c_6	$4 \cdot \|\mathcal{L}\| \cdot \|\mathcal{R}\|$	$4 \cdot c_6 \cdot \|\mathcal{L}\| \cdot \|\mathcal{R}\|$
8	c_7	$4 \cdot \|\mathcal{L}\| \cdot \|\mathcal{R}\|$	$4 \cdot c_7 \cdot \|\mathcal{L}\| \cdot \|\mathcal{R}\|$
9	$c_8 \cdot \|\mathcal{L}\|$	$4 \cdot \|\mathcal{L}\| \cdot \|\mathcal{R}\|$	$4 \cdot c_8 \cdot \|\mathcal{L}\|^2 \cdot \|\mathcal{R}\|$
10	c_9	$4 \cdot \|\mathcal{L}\| \cdot \|\mathcal{R}\|$	$4 \cdot c_9 \cdot \|\mathcal{L}\| \cdot \|\mathcal{R}\|$
11	$c_{10} \cdot \|\mathcal{R}\|$	$4 \cdot \|\mathcal{R}\|$	$4 \cdot c_{10} \cdot \|\mathcal{R}\|^2$
12	c_{11}	$4 \cdot \|\mathcal{R}\|$	$4 \cdot c_{11} \cdot \|\mathcal{R}\|$
13	$c_{12} \cdot \|\mathcal{R}\|$	$4 \cdot \|\mathcal{L}\|$	$4 \cdot c_{12} \cdot \|\mathcal{L}\| \cdot \|\mathcal{R}\|$
14	$c_{13} \cdot \|\mathcal{R}\|$	$4 \cdot \|\mathcal{R}\|$	$4 \cdot c_{13} \cdot \|\mathcal{R}\|^2$
15	c_{14}	$4 \cdot \|\mathcal{R}\|$	$4 \cdot c_{14} \cdot \|\mathcal{R}\|$
16	$c_{15} \cdot \|\mathcal{R}\|$	$4 \cdot \|\mathcal{L}\|$	$4 \cdot c_{15} \cdot \|\mathcal{L}\| \cdot \|\mathcal{R}\|$
Total time required for all lines			$\mathcal{O}(\|\mathcal{L}\|^2 \cdot \|\mathcal{R}\| + \|\mathcal{R}\|^2)$

Table 1: Complexity per line of Algorithm 1.

the labels of all rules for that literal. Line 6 requires $c_5 \cdot |\mathcal{R}|$ time per execution, since a literal is antecedent of at most $|\mathcal{R}|$ rules. Line 7 only needs to check if a set is empty, which can be done in constant time c_6. The next line only needs to pop an element from a set, which can be done in constant time as well, so line 8 needs c_7 time. Line 9 relabels a rule, which requires checking the labels of all antecedents of this rule. Since a rule has at most $|\mathcal{L}|$ antecedents, this takes at most $c_8 \cdot |\mathcal{L}|$ time per execution of Line 9. Lines 10, 12 and 15 only check if the label of a rule or literal changed; this can be done in constant time c_9, c_{11} and c_{14} respectively. Lines 11 and 14 relabel a literal, which requires checking the labels of all rules for that literal. A single execution therefore takes $c_{10} \cdot |\mathcal{R}|$ time for 11 and $c_{13} \cdot |\mathcal{R}|$ time for 14. Finally, lines 13 and 16 both add at most $|\mathcal{R}|$ rules to TODO-SET, so a single execution of line 13 needs at most $c_{12} \cdot |\mathcal{R}|$ time and a single execution of line 16 needs at most $c_{15} \cdot |\mathcal{R}|$ time.

Now we consider the number of iterations of each line; this is also represented in the third column of Table 1. Lines 2 and 3 are executed just once. Lines 4–6 are repeated for each literal in $\mathcal{Q}$, which must be in $\mathcal{L}$ and for each rule in $\mathcal{R}$. This implies that lines 4–6 are executed at most $|\mathcal{L}| + |\mathcal{R}|$ times. The lines 7–10 are executed in every iteration of the while loop. The total number of iterations of the while loop equals the number of times a rule is added to TODO-SET. A rule is only added to TODO-SET if it was not yet visited (line 6) or if the label of one of its antecedents changed after a relabelling (line 13 or line 16). Since the label of a literal can change at most three times (i.e. at most three booleans can be turned to False), each rule r is recolored at most $4 \cdot |\texttt{ants}(r)|$ times. There are $|\mathcal{R}|$ rules, so lines 7–10 are executed at most $4 \cdot |\mathcal{L}| \cdot |\mathcal{R}|$ times. Next, we consider lines 11, 12, 14 and 16. These lines are only executed if the label of a rule changed. A label can only be changed by turning one of the four booleans to False. Therefore, a label can be changed at most four times for each rule (we will see in Lemma 10 that in practice, the maximum of changes is three). There are $|\mathcal{R}|$ rules in total, so lines 11, 12, 14 and 16 are executed at most $4 \cdot |\mathcal{R}|$ times. Finally, lines 13 and 16 are only executed just after the label of a literal changed. This can happen at most four times for each literal, because at most four booleans can be turned to False. (Again, we will see in Lemma 10 that the maximum of changes per label is in practice 3.) There are $|\mathcal{L}|$ literals in total; therefore lines 13 and 16 are executed at most $4 \cdot |\mathcal{L}|$ times.

An upper bound on the total amount of time that is needed for all executions of a single line can now be obtained by multiplying the maximum time required for a single execution by the number of executions of each line. We do this in the fourth column of Table 1. From these results, it becomes clear that the total running time of Algorithm 1 is dominated by the lines for relabeling: line 9 takes $4 \cdot c_8 \cdot |\mathcal{L}|^2 \cdot |\mathcal{R}|$ time; line 11 takes $4 \cdot c_{10} \cdot |\mathcal{R}|^2$ time and line 14 takes $4 \cdot c_{13} \cdot |\mathcal{R}|^2$ time. To conclude, the total time complexity of Algorithm 1 is $\mathcal{O}(|\mathcal{L}|^2 \cdot |\mathcal{R}| + |\mathcal{R}|^2)$. □

3.3 Preprocessing

The new labelling proposed in the previous section does not solve the support cycle problem: if we would apply the labelling L' from Definition 9 to the argumentation setup from Figure 4, all literals l (including topic literal t) would be labelled $\langle 1,1,1,1\rangle$. In order to solve this issue, we add a preprocessing step, which is specified in Algorithm 2. The idea of this algorithm is that initially, all literals that cannot be in the knowledge base in a future setup and all rules are labelled $\langle 1,0,0,0\rangle$ (i.e. unsatisfiable). Then, the algorithm incrementally removes unsatisfiable labels of rules for which all antecedents are not labelled $\langle 1,0,0,0\rangle$, and of the consequents of these rules, based on the intuition that there may be an argument based on these rules in a future setup.

Algorithm 2 Preprocessing step (to obtain L_p)

```
1: procedure PREPROCESS(L, R, Q, K)
2:     Label each literal l s.t. l ∈ Q ∧ −l ∉ K as ⟨1,1,1,1⟩
3:     Label all other literals as ⟨1,0,0,0⟩
4:     Label each r ∈ R as ⟨1,0,0,0⟩
5:     while a label changed in the previous loop do
6:         for Rule r in R do
7:             if L(r) = ⟨1,0,0,0⟩ and for each l ∈ ants(r): L(l) ≠ ⟨1,0,0,0⟩ then
8:                 Label r as ⟨1,1,1,1⟩
9:                 Label cons(r) as ⟨1,1,1,1⟩
```

Lemma 7 (Time complexity preprocessing)**.** *The time complexity of Algorithm 2 is* $\mathcal{O}(|\mathcal{L}|\cdot|\mathcal{R}|^2)$.

Proof. Algorithm 2 needs at most $c_1\cdot(|\mathcal{L}|+|\mathcal{R}|)$ operations for line 2–4, where c_1 is a positive constant. Next we consider the time required for the lines in the while-loop. We stay in the while-loop until no label changed any more in the previous loop, so at least one label had to change in the previous loop. Thanks to the check $L(r)=\langle 1,0,0,0\rangle$, for each rule the label can change at most once. Therefore the while-loop iterates at most $|\mathcal{R}|$ times. Line 5 only requires a label check, which can be done in constant time c_2. The for-loop iterates $|\mathcal{R}|$ times for each iteration of the while-loop. As a result, the lines 6–9 are executed at most $|\mathcal{R}|^2$ times in total. A single execution of line 6 takes constant time c_3. Line 7 checks all antecedents of each rule, which requires at most $c_4\cdot|\mathcal{L}|$ checks per execution where c_4 is a positive constant; line 8 and 9 take constant time c_5. So the total time required for all executions of lines 6–9 is at most $(c_3+c_4\cdot|\mathcal{L}|+c_5)\cdot|\mathcal{R}|^2$. The total time required for Algorithm 2 is $c_1+c_2\cdot|\mathcal{R}|+(c_3+c_4\cdot|\mathcal{L}|+c_5)\cdot|\mathcal{R}|^2$ time. As a result, the time complexity of the preprocessing step must be $\mathcal{O}(|\mathcal{L}|\cdot|\mathcal{R}|^2)$. □

Lemma 8 (Soundness preprocessing step)**.** *Given an argumentation setup* $AS=(\mathcal{L},\mathcal{R},\mathcal{Q},\mathcal{K})$ *and labelling* L_p *after the preprocessing step, for each* $l\in\mathcal{L}$*: if* $L_p(l)=\langle 1,0,0,0\rangle$*, then for each* $AS'\in F(AS)$*:* l *is unsatisfiable in* AS'.

Proof. We prove this by contraposition. Let $l\in\mathcal{L}$ be a literal and let L_p be the labelling after the preprocessing step. Furthermore suppose that not for each $AS'\in F(AS)$: l is unsatisfiable in AS'. By Definition 6, implies that there exists an $AS'=(\mathcal{L},\mathcal{R},\mathcal{Q},\mathcal{K}')$ in $F(AS)$ such that there is an argument for l in $Arg(AS')$. By Lemma 1 there is a non-circular argument A for l in $Arg(AS')$. We now prove by induction on the height of A that l is labelled $\langle 1,1,1,1\rangle$ by Algorithm 2.

Base case: Suppose that $\mathtt{h}(A)=0$. Then $l\in\mathcal{K}'$. This implies that $l\in\mathcal{Q}\wedge -l\notin\mathcal{K}$, so l is labelled $\langle 1,1,1,1\rangle$ in Algorithm 2 line 2. Since there is no operation which labels literals from $\langle 1,1,1,1\rangle$ to $\langle 1,0,0,0\rangle$, we have that $L_p(l)=\langle 1,1,1,1\rangle$.

Induction hypothesis: If A is a non-circular argument in $Arg(AS')$ and $\mathtt{h}(A)\leq k$, then $L_p(\mathtt{conc}(A))=\langle 1,1,1,1\rangle$.

Induction step: Now suppose that $\mathtt{h}(A)=k+1$. Then A must be a rule-based non-circular argument $A_1,\ldots,A_m\Rightarrow l$ in $Arg(AS')$. This implies that for each $i\in[1\,..\,m]$: A_i in $Arg(AS')$ and $\mathtt{h}(A_i)\leq k$. By the induction hypothesis, for each $a=\mathtt{ants}(\mathtt{top\text{-}rule}(A))$: $L_p(a)=\langle 1,1,1,1\rangle$, so $L_p(l)$ is labelled $L_p(l)=\langle 1,1,1,1\rangle$ by Algorithm 2 line 9. Since there is no operation which labels literals from $\langle 1,1,1,1\rangle$ to $\langle 1,0,0,0\rangle$, we have that $L_p(l)=\langle 1,1,1,1\rangle$.

We just proved that each literal $l \in \mathcal{L}$ for which there is an $AS' \in F(AS)$ such that there is an argument for l in $Arg(AS')$ is labelled $L_p(l) = \langle 1,1,1,1 \rangle$ (so $L_p(l) \neq \langle 1,0,0,0 \rangle$). This implies: if $L_p(l) = \langle 1,0,0,0 \rangle$, then for each $AS' \in F(AS)$: l is unsatisfiable in AS'. □

Example 9 (Alternative labelling Example 6)**.** Let us reconsider Figure 4, assuming that the preprocessing step has been executed. In Line 3, all literals (a, b, c and t) are labelled $\langle 1,0,0,0 \rangle$. Since the if-statement in Line 7 never returns true, no rule or literal gets another label, so the while loop is executed only once. After termination of Algorithm 2, all literals are still (correctly) labelled $\langle 1,0,0,0 \rangle$.

Algorithm 3 Stability labeling

1: **procedure** STABILITY($\mathcal{L}, \mathcal{R}, \mathcal{Q}, \mathcal{K}$)
2: Apply PREPROCESS($\mathcal{L}, \mathcal{R}, \mathcal{Q}, \mathcal{K}$) (Algorithm 2) to obtain L_p for each literal/rule x in $\mathcal{L}$ and $\mathcal{R}$
3: Apply LABELLING-PROCEDURE($\mathcal{L}, \mathcal{R}, \mathcal{Q}, \mathcal{K}$) (Algorithm 1) line 3–16

Finally, Algorithm 3 shows our proposed algorithm STABILITY. The algorithm runs PREPROCESS on the argumentation setup and then labels all literals and rules by repeatedly applying Definition 9.

3.4 Properties of the proposed algorithm

In this subsection, we present properties of STABILITY, our approximation algorithm for estimating stability.

3.4.1 Soundness

STABILITY is sound: if the algorithm labels a literal l as stable in an argumentation setup AS, then l is stable in AS. We will formally prove this in Proposition 2. In order to prove this, we first need to prove the following lemmas for a given argumentation setup $AS = (\mathcal{L}, \mathcal{R}, \mathcal{Q}, \mathcal{K})$:

Lemma 9 Each rule $r \in \mathcal{R}$ is labelled $\neg u(r)$ iff there is an argument based on r in $Arg(AS)$;

This lemma explains the behaviour of the boolean representing the unsatisfiablility for rules. That boolean is turned to false if and only if an argument based on that rule can be inferred from the current argumentation setup. In this case, it is guaranteed that there is an argument for all antecedents of this rule in the current argumentation setup. Therefore, all antecedents of the rule are certainly not unsatisfiable in each future argumentation setup.

Lemma 10 No literal $l \in \mathcal{L}$ or rule $r \in \mathcal{R}$ is labelled $\langle 0,0,0,0 \rangle$;

This lemma excludes the possibility that all four acceptability booleans (unsatisfiable, defended, out and blocked) are turned to false by the labeling algorithm. This is an important property that we will use repeatedly to show that a literal or rule is stable after proving that three of its acceptability booleans have been turned to False.

Lemma 11 If a rule $r \in \mathcal{R}$ for l is labelled $\neg u(r)$ and $\neg o(r)$ and $l \notin \mathcal{Q}$, then l is not unsatisfiable or out in any future argumentation setup of AS;

This lemma analyses the situation that both the unsatisfiable and the out boolean for a rule are turned to False, while its consequent (as well as the negation of the consequent) is not observable. In this situation, there must be some argument based on that rule in each future setup AS' that is not attacked by an argument in the grounded extension $G(AS')$.

Lemma 12 If a rule $r \in \mathcal{R}$ for l is labelled $\neg d(r)$ and $\neg b(r)$ then l cannot be defended or blocked in any future argumentation setup of *AS thanks to that rule*.

If both the defended and out case of a rule r are turned false, then there is no future argumentation setup AS' in which there is an argument based on r that is not attacked by an argument in the grounded extension of AS'.

These lemmas will be used repeatedly in the proof of Proposition 2 (soundness); furthermore Lemma 9 right-to-left is used in Proposition 3 (conditional completeness). Remember that a literal l is labelled stable by L' iff $L'(l) = \langle 1,0,0,0\rangle$, $L'(l) = \langle 0,1,0,0\rangle$, $L'(l) = \langle 0,0,1,0\rangle$ or $L'(l) = \langle 0,0,0,1\rangle$. The stability status of a literal l is in many cases dependent on the stability statuses of the rules for l. For example, if we want to show that the *out* case is sound, then we need to prove: if $L'(l) = \langle 0,0,1,0\rangle$ then for each $AS' \in F(AS)$, l is out in AS'. $L'(l) = \langle 0,0,1,0\rangle$ iff l is labelled $\neg u(l)$, $\neg d(l)$ and $\neg b(l)$ by L' and $L'(l) \neq \langle 0,0,0,0\rangle$. Lemma 10 excludes the possibility that $L'(l) = \langle 0,0,0,0\rangle$. We will use Lemma 9 to show that there is an argument for l in $Arg(AS)$; which must be in $Arg(AS')$ for each $AS' \in F(AS)$, so l is not unsatisfiable in any $AS' \in F(AS)$. In case $l \notin \mathcal{Q}$, we will use Lemma 12 to prove that l is not defended or blocked in any $AS' \in F(AS)$ - which implies that l is out in each $AS' \in F(AS)$. In a similar way, Lemma 11 will be used in the soundness proof for the blocked case.

Lemma 9 (Argument existence labelling)**.** *Given an argumentation setup $AS = (\mathcal{L}, \mathcal{R}, \mathcal{Q}, \mathcal{K})$, let L' be the labelling obtained by* STABILITY *(Algorithm 3). For each $r \in \mathcal{R}$: r is labelled $\neg u(r)$ by L' iff there is an argument based on r in $Arg(AS)$.*

Proof. Let $AS = (\mathcal{L}, \mathcal{R}, \mathcal{Q}, \mathcal{K})$ be an argumentation setup and let L' be the labelling obtained by STABILITY (Algorithm 3). Furthermore let r be an arbitrary rule in $\mathcal{R}$. We will prove that r is labelled $\neg u(r)$ by L' iff there is an argument based on r in $Arg(AS)$ in both directions.

$\rightarrow$: if r is labelled $\neg u(r)$ by L', then there is an argument based on r in *Arg*(AS).

r cannot be labelled $\neg u(r)$ in the preprocessing step (Algorithm 2: PREPROCESS), since none of the operations in Algorithm 2 turns the u-boolean of any rule to False. The u-boolean of a rule can only be relabelled by Algorithm 1 (LABELLING-PROCEDURE) line 9. We now prove that there is an argument based on r in $Arg(AS)$ by induction on the number of the iteration of the while loop in which the u-boolean of r was turned from True to False.

Base case: First suppose that r was labelled $\neg u(r)$ in the first iteration of the while loop. Then by Definition 9 case R-U-a, for each $a \in \texttt{ants}(r) : \neg u(a)$. Each antecedent $a \in \texttt{ants}(r)$ must have been labelled $\neg u(a)$ by Algorithm 1 line 5. Since each rule $r' \in \mathcal{R}$ is labelled $u(r')$ before the first iteration of the while loop, $\neg u(a)$ cannot be caused by case L-U-b, so must be caused by L-U-a: for each $a \in \texttt{ants}(r) : a \in \mathcal{K}$. So by Definition 2, for each $a \in \texttt{ants}(r)$ there is an argument for a in $Arg(AS)$; therefore, there is an argument based on r in $Arg(AS)$.

Induction hypothesis: If r was labelled $\neg u(r)$ in the k'th iteration of the while loop or earlier, then there is an argument based on r in $Arg(AS)$.

Induction step: Now suppose that r was labelled $\neg u(r)$ in the $(k+1)$'th iteration of the while loop (by Algorithm 1 line 9). Consider an arbitrary antecedent $a \in \texttt{ants}(r)$. a must have been labelled $\neg u(a)$ in the k'th iteration of the while loop or earlier (Definition 9 case R-U-a). So either $a \in \mathcal{K}$ (case L-U-a) or for each r' for a: r' is labelled $\neg u(r')$ in or before the k'th iteration of the while loop (case L-U-b). If $a \in \mathcal{K}$, then there is an observation-based argument for a in $Arg(AS)$. Otherwise, there is a rule r' that is labelled $\neg u(r')$ in the k'th iteration of the while loop or earlier, so by the induction hypothesis there is an argument based on r' in $Arg(AS)$. $\texttt{cons}(r') = a$, so there is an argument for a in $Arg(AS)$. Since we picked a arbitrarily from the antecedents of r, we have that for each $a \in \texttt{ants}(r)$, there is an argument for a in $Arg(AS)$. So by definition of rule-based arguments, there is an argument based on r in $Arg(AS)$.

$\leftarrow$: if there is an argument based on r in *Arg*(AS), then r is labelled $\neg u(r)$ by L'.

Suppose that there is an argument based on r in $Arg(AS)$. By Definition 2, for each $a \in \texttt{ants}(r)$, there is an argument for a in $Arg(AS)$. Now let a be an arbitrary literal such that $a \in \texttt{ants}(r)$. By Lemma 1, there is a non-circular argument A for a in $Arg(AS)$. We prove that a is labelled $\neg u(a)$ by induction on the height of A.

Base case: First suppose that $\texttt{h}(A) = 0$. Then $a \in \mathcal{K}$, so $a \in \mathcal{Q}$. This means that the condition for the if-statement in Algorithm 1 line 4 applies and a is labelled in Algorithm 1 line 5 and will be labelled $\neg u(a)$ by case L-U-a.

Induction hypothesis: If A is a non-circular argument for a with $\mathtt{h}(A) \leq k$, then a is labelled $\neg u(a)$ by L'.

Induction step: Now suppose that $\mathtt{h}(A) = k+1$. So A has the form $A_1, \ldots, A_m \Rightarrow a$ and for each $i \in [1 \,..\, m] : A_i \in Arg(AS)$. Let $\mathtt{top\text{-}rule}(A) = r'$. For each $i \in [1 \,..\, m]$, we know that $\mathtt{h}(A_i) \leq k$ and A_i is non-circular (since A is non-circular), so we can apply the induction hypothesis: for each $a' \in \mathtt{ants}(r')$, a' is labelled $\neg u(a')$ by L'. Next, we show that r' is considered for relabelling by Algorithm 1. Let a'_l be the last antecedent of r' that was labelled $\neg u(a'_l)$. a'_l is relabelled to $\neg u(a'_l)$ either in Algorithm 1 line 5, line 11 or line 14. In all cases, r' is added to TODO-SET immediately afterwards (by either line 6, line 13 or line 16, so r' is considered for relabelling after all antecedents a' of r' were labelled. Given that r' is added to TODO-SET, it must be popped from this set before termination of Algorithm 1. Then r' is relabelled by Algorithm 1 line 9. By Definition 9 case R-U-a, r' is labelled $\neg u(r')$. After this, $\mathtt{cons}(r') = a$ was relabelled in Algorithm 1 line 11. By Definition 9 case L-U-B, a was labelled $\neg u(a)$.

We showed by induction that each literal $a \in \mathtt{ants}(r)$ is labelled $\neg u(a)$ by L'. Let a_l the last antecedent of r which was labelled $\neg u(a_l)$. a_l is relabelled to $\neg u(a_l)$ either in Algorithm 1 line 5, line 11 or line 13. In all cases, r is added to TODO-SET immediately afterwards, so r is considered for relabelling (i.e. added to TODO-SET) after all antecedents a of r were labelled. Given that r is added to TODO-SET, it must be popped from this set before termination of Algorithm 1 and relabelled in line 9. By Definition 9 case R-U-a, r is labelled $\neg u(r)$.

We proved that r is labelled $\neg u(r)$ by L' iff there is an argument based on r in $Arg(AS)$. Since we chose r as an arbitrary rule in $\mathcal{R}$, this goes for every rule r in $\mathcal{R}$. □

Lemma 10 (No 4x False label)**.** *Given an argumentation setup $AS = (\mathcal{L}, \mathcal{R}, \mathcal{Q}, \mathcal{K})$, let L' be the labelling obtained by* STABILITY *(Algorithm 3). For each $x \in \mathcal{L} \cup \mathcal{R}$: $L'(x) \neq \langle 0,0,0,0 \rangle$.*

Proof. We first consider the situation that x is labelled unsatisfiable in the preprocessing step: $L_p(x) = \langle 1,0,0,0 \rangle$.

- Let l be a literal ($x \in \mathcal{L}$, $x = l$) and suppose that $L_p(l) = \langle 1,0,0,0 \rangle$. Then by Lemma 8, for each $AS' \in F(AS)$: l is unsatisfiable in AS'. $AS \in F(AS)$ (Definition 7), so l is unsatisfiable in AS. By Definition 6, this implies that there is no argument for l in AS. Now we prove by contradiction that $L'(l) \neq \langle 0,0,0,0 \rangle$. Suppose that $L' = \langle 0,0,0,0 \rangle$. By Definition 9, this must have been caused by either $l \in \mathcal{K}$ (case L-U-a) or because there is a rule r for l which is labelled $\neg u(r)$ (case L-U-b). However, $l \notin \mathcal{K}$ because there is no argument for l in $Arg(AS)$. If there would be a rule r for l which is labelled $\neg u(r)$, then by Lemma 9, there is an argument based on r in $Arg(AS)$, which again contradicts our assumption that there is no argument for l. So if l is labelled unsatisfiable in the preprocessing step (i.e. $L_p(l) = \langle 1,0,0,0 \rangle$), then $L'(l) \neq \langle 0,0,0,0 \rangle$.

- Let r be a rule ($x \in \mathcal{R}, x = r$). Suppose that $L_p(r) = \langle 1,0,0,0 \rangle$. This implies that there is an antecedent $a \in \mathtt{ants}(r) : L_p(a) = \langle 1,0,0,0 \rangle$: otherwise, the label of r would have been changed to $\langle 1,1,1,1 \rangle$ by Algorithm 2 line 8. So by Lemma 8, there is an antecedent $a \in \mathtt{ants}(r)$ such that for each $AS' \in F(AS)$: a is unsatisfiable in AS'. Consequently, there is no argument for a in $Arg(AS)$, so there is no argument based on r in $Arg(AS)$. Then by Lemma 9, r is not labelled $\neg u(r)$ by L'. To conclude: if r is labelled unsatisfiable in the preprocessing step (i.e. $L_p(r) = \langle 1,0,0,0 \rangle$), then $L'(r) \neq \langle 0,0,0,0 \rangle$.

Alternatively, assume that x is not labelled $\langle 1,0,0,0 \rangle$ in the preprocessing step: $L_p(x) = \langle 1,1,1,1 \rangle$. Let $c(x)$ be the number of the iteration of the while loop (Algorithm 1 line 7–16) in which x is relabelled for the last time; so if x was relabelled for the last time after the first iteration of the while loop, then $c(x) = 1$. Furthermore, let L_i be the labelling immediately after the i'th iteration of the while loop. We show that $L'(x) \neq \langle 0,0,0,0 \rangle$ by induction on $c(x)$.

Base case: First assume that $c(x) = 0$.

- Let r be a rule ($x \in \mathcal{R}$, $x = r$). We have that $L_p(r) = \langle 1,1,1,1 \rangle$ by an earlier assumption. No rule is relabelled before the first iteration of the while loop, so given that $c(r) = 0$, we have $L'(r) = \langle 1,1,1,1 \rangle \neq \langle 0,0,0,0 \rangle$.

- Alternatively, let l be a literal ($x \in \mathcal{L}$, $x = l$). By an earlier assumption, $L_p(l) = \langle 1,1,1,1 \rangle$. This can only be caused by the fact that $(l \in \mathcal{Q} \wedge -l \notin \mathcal{K})$ (Algorithm 2 line 2) or there is a rule r for l such that $L_p(r) = \langle 1,1,1,1 \rangle$ (Algorithm 2 line 9). We will show that in both cases l is labelled $d(l)$.
  - First assume that $l \in \mathcal{Q}$ and $-l \notin \mathcal{K}$. This implies that l is considered for relabelling in Algorithm 1 line 5. In this relabelling step, none of the labelling rules L-D-a/b/c from Definition 9 applies, so l is labelled $d(l)$.
  - Alternatively, there is a rule r for l such that $L_p(r) = \langle 1,1,1,1 \rangle$. Rules are not relabelled before the first iteration of the while loop, so there is a rule r for l such that $L_0(r) = L_p(r) = \langle 1,1,1,1 \rangle$. Furthermore, for each rule $r \in \mathcal{R} : u(r)$ since the u-boolean of rules cannot be turned False before the first iteration of the while loop. As a result, none of the labelling rules L-D-a/b/c from Definition 9 applies; therefore l is labelled $d(l)$.

  To conclude, in both cases l is labelled $d(l)$. Therefore, $L'(l) \neq \langle 0,0,0,0 \rangle$.

So for each $x \in \mathcal{L} \cup \mathcal{R}$ such that $c(x) = 0$: $L'(x) \neq \langle 0,0,0,0 \rangle$.
Induction hypothesis: For each $x \in \mathcal{L} \cup \mathcal{R}$ such that $c(x) \leq k$: $L'(x) \neq \langle 0,0,0,0 \rangle$.
Induction step:

- First, we consider rules. Let $r \in \mathcal{R}$ be a rule such that $c(r) = k+1$. This implies that for each antecedent $a \in \texttt{ants}(r) : c(a) \leq k$: rules can only be relabelled by Algorithm 1 line 9, so they must have been added to TODO-SET after the labelling of an antecedent changed in a previous iteration of the while loop.

  We will prove by contradiction that $L'(r) \neq \langle 0,0,0,0 \rangle$. Suppose that $L'(r) = \langle 0,0,0,0 \rangle$. Then $\neg d(r)$, which must be caused by Definition 9 case R-D-a, so there is an antecedent $a \in \texttt{ants}(r) : \neg d(a)$. Furthermore, we know that $\neg b(r)$. This can be caused by either R-B-a or R-B-b, but in both cases, there is an antecedent $a \in \texttt{ants}(r)$ such that $\neg d(a)$ and $\neg b(a)$. Similarly, the fact that r is labelled $\neg o(r)$ can be caused by either R-O-a or R-O-b, but in both cases, there is an antecedent $a \in \texttt{ants}(r)$ such that $\neg d(a)$ and $\neg o(a)$ and $\neg b(a)$. Finally, $\neg u(r)$ implies that $\neg u(a)$ for each antecedent a of r. But then there is an antecedent $a \in \texttt{ants}(r)$ such that $\neg u(a)$, $\neg d(a)$, $\neg o(a)$ and $\neg b(a)$: $L'(a) = \langle 0,0,0,0 \rangle$. However, this contradicts our induction hypothesis; therefore, there is no rule $r \in \mathcal{R}$ such that $c(r) = k+1$ and $L'(r) = \langle 0,0,0,0 \rangle$.

- Now we consider literals: let $l \in \mathcal{L}$ be a literal such that $c(l) = k+1$. Note that $c(l) = k+1$ implies that l was relabelled for the last time within (the $k+1$'th iteration of) the while loop. Therefore, l must have been relabelled for the last time by Algorithm 1 in either line 11 or line 14. This implies that each rule for l or $-l$ must have been labelled in iteration $k+1$ or earlier. Formally: for each rule r such that $\texttt{cons}(r) \in \{l, -l\}$: $c(r) \leq k+1$. From the induction hypothesis (there is no rule $r \in \mathcal{R}$ such that $c(r) \leq k$ and $L'(r) = \langle 0,0,0,0 \rangle$) and our earlier conclusion that there is no rule $r \in \mathcal{R}$ such that $c(r) = k+1$ and $L'(r) = \langle 0,0,0,0 \rangle$, we derive: there is no rule $r \in \mathcal{R}$ such that $c(r) \leq k+1$ and $L'(r) = \langle 0,0,0,0 \rangle$.

  We will now prove by contradiction that there is no literal $l \in \mathcal{L}$ such that $c(l) = k+1$ and $L'(l) = \langle 0,0,0,0 \rangle$. Suppose that $L'(l) = \langle 0,0,0,0 \rangle$.

  First suppose that $l \in \mathcal{Q}$. This implies that $\neg d(l)$ must have been caused by Definition 9 case L-D-a, so $-l \in \mathcal{K}$. Then by Definition 1, $l \notin \mathcal{K}$, so $\neg u(l)$ must have been caused by L-U-b: there is a rule r for l with $\neg u(r)$. Furthermore, $\neg o(l)$ must have been caused by L-O-b: for each rule r for l: $\neg d(r)$, $\neg o(r)$ and $\neg b(r)$. But that implies that there exists a rule r for l such that $L'(r) = \langle 0,0,0,0 \rangle$. However, this contradicts our earlier conclusion that there is no rule r for l such that $L'(r) = \langle 0,0,0,0 \rangle$. So $l \notin \mathcal{Q}$.

  Given that $l \notin \mathcal{Q}$, $\neg u(l)$ must have been caused by case L-U-b: there is a rule r for l with $\neg u(r)$. Then $\neg o(l)$ cannot be caused by L-O-a (since $l \notin \mathcal{K}$) or L-O-b (since there is no rule r for l such that and $L'(r) = \langle 0,0,0,0 \rangle$). $\neg o(l)$ must be caused by either L-O-c or L-O-d; in both cases, there is a rule r for l with $\neg u(r)$ and $\neg o(r)$. This implies that $\neg b(l)$ cannot be caused by L-B-b, so it must have been caused by L-B-c or L-B-d. In both cases, there is a rule r for l with $\neg u(r)$, $\neg o(r)$ and

$\neg b(r)$ and for each rule r' for $-l$: $\neg d(r')$ and $\neg b(r')$. Finally, $\neg d(l)$ can be caused by either L-D-b or L-D-c. However, both cases contradict the induction hypothesis: if it was caused by L-D-b, then there is a rule r for l with $L(r) = \langle 0,0,0,0\rangle$; if it was caused by L-D-c, then there is a rule r' for $-l$ with $L(r') = \langle 0,0,0,0\rangle$.

To conclude, for each $l \in \mathcal{L}$ such that $c(l) = k+1$ we have that $L'(l) \neq \langle 0,0,0,0\rangle$. □

Lemma 11 (Labelled not unsatisfiable or out)**.** *Given an argumentation setup $AS = (\mathcal{L}, \mathcal{R}, \mathcal{Q}, \mathcal{K})$, let L' be the labelling obtained by* STABILITY *(Algorithm 3). For each $r \in \mathcal{R}$: if $\texttt{cons}(r) \notin \mathcal{Q}$ and r is labelled $\neg u(r)$ and $\neg o(r)$ by L', then for each $AS' \in F(AS)$, there is an argument based on r in $Arg(AS')$ that is not attacked by any argument in $G(AS')$.*

Proof. Suppose that r is a rule in $\mathcal{R}$ such that $\texttt{cons}(r) \notin \mathcal{Q}$ and r is labelled $\neg u(r)$ and $\neg o(r)$ by L'. This implies that r is not labelled $\langle 1,0,0,0\rangle$ in the preprocessing step (otherwise the fact that r is labelled $\neg u(r)$ would imply that $L'(r) = \langle 0,0,0,0\rangle$, which contradicts Lemma 10). So $L_p(r) = \langle 1,1,1,1\rangle$. Let $c(r)$ be the number of the iteration of the while loop (Algorithm 1 line 7–16) in which r is relabelled for the last time and let L_i be the labelling immediately after the i'th iteration of the while loop. We show by induction on $c(r)$ that for each $AS' \in F(AS)$, there is an argument based on r in $Arg(AS')$ that is not attacked by any argument in $G(AS')$.

We choose $c(r) = 1$ as base case: if $c(r) = 0$ then r is not relabelled in the while-loop. This implies that $L'(r) = L_p(r) = \langle 1,1,1,1\rangle$, which contradicts our assumption that r is labelled $\neg u(r)$ and $\neg o(r)$ by L'.

Base case: If $c(r) = 1$, then r is relabelled for the last time in the first iteration of the while-loop. By Definition 9, the fact that r is labelled $\neg u(r)$ must have been caused by case R-U-a. Then by Lemma 10 the fact that r is labelled $\neg o(r)$ must have been caused by case R-O-a. Given that $c(r) = 1$, r must have been added to TODO-SET in Algorithm 1 line 6. So for each $a \in \texttt{ants}(r)$: $a \in \mathcal{Q}$ or there is no rule for a in $\mathcal{R}$. Next, we show that each $a \in \texttt{ants}(r)$ is in $\mathcal{K}$. Let a be an arbitrary antecedent of r. We consider both the case that there is a rule for a in $\mathcal{R}$ and the case that there is no such rule.

- Suppose that there is a rule r' for a in $\mathcal{R}$. r' is relabelled for the last time *before* the first iteration of the while-loop. Then r' must be labelled $u(r')$: before the first iteration of the while-loop, there is no line that turns the u-boolean to False. Then the fact that a is labelled $\neg u(a)$ cannot be caused by Definition 9 case L-U-b, so it must have been caused by case L-U-a: $a \in \mathcal{K}$.

- Alternatively, there is no rule r' for a in $\mathcal{R}$. Then the fact that a is labelled $\neg u(a)$ cannot be caused by Definition 9 case L-U-b, so it must have been caused by case L-U-a: $a \in \mathcal{K}$.

We chose a arbitrarily, so for each $a \in \texttt{ants}(r)$: $a \in \mathcal{K}$. Now let A be the argument based on r and these observation-based arguments for the antecedents of r. Then for each $a \in \texttt{ants}(r)$: for each $AS' \in F(AS)$ there is an argument for a in $Arg(AS')$ that cannot be attacked, so it is not attacked by an argument in $G(AS')$. This means that for each $AS' \in F(AS)$, A is not attacked *on a subargument* in AS'. Furthermore, note that $\texttt{cons}(r) \notin \mathcal{Q}$. This means that there is no observation-based argument for $-\texttt{cons}(r)$ in any future argumentation setup. By Lemma 4, for each $AS' \in F(AS)$: A is attacked by an argument in the grounded extension $G(AS')$ iff A is attacked by an observation-based argument. Therefore A cannot be attacked *on its conclusion* by an argument in $G(AS')$. To conclude, for each $AS' \in F(AS)$ there is an argument (A) based on r in $Arg(AS')$ that is not attacked by any argument in $G(AS')$.

Induction hypothesis: For each $r \in \mathcal{R}$: if $\texttt{cons}(r) \notin \mathcal{Q}$, $c(r) \leq k$ and r is labelled $\neg u(r)$ and $\neg o(r)$ by L', then for each $AS' \in F(AS)$, then there is an argument based on r in $Arg(AS')$ that is not attacked by any argument in $G(AS')$.

Induction step: Now suppose that $c(r) = k+1$. This implies that for each antecedent $a \in \texttt{ants}(r)$: $c(a) \leq k$: rules can only be relabelled by Algorithm 1 line 9, so they must have been added to TODO-SET after the labelling of an antecedent changed in a previous iteration of the while loop. By Definition 9 case R-U-a, each antecedent a of r is labelled $\neg u(a)$. As a result, the fact that $\neg o(r)$ must be caused by case R-O-a, so each antecedent a of r is labelled $\neg u(a)$ and $\neg o(a)$. Let a be an arbitrary antecedent in $\texttt{ants}(r)$. We now consider two cases: either $a \in \mathcal{K}$ of $a \notin \mathcal{K}$.

If $a \in \mathcal{K}$, then a was labelled $\neg u(a)$ by L-U-a and $\neg o(a)$ by L-O-a. By Definition 2, for each $a \in \texttt{ants}(r)$, there is an argument for a in $Arg(AS)$, so there is an argument for $\texttt{cons}(r)$ based on r in $Arg(AS)$.

By Lemma 5, this argument is in $Arg(AS')$ for every $AS' \in F(AS)$. Since $\texttt{cons}(r) \notin \mathcal{Q}$ and for each antecedent a in $\texttt{ants}(r)$: $a \in \mathcal{K}$, the argument based on r cannot be attacked by an observation-based argument in any $AS' \in F(AS)$ (Definition 4). So by Lemma 4, for each $AS' \in F(AS)$, there is an argument based on r in $Arg(AS')$ that is not attacked by any argument in $G(AS')$.

Alternatively, $a \notin \mathcal{K}$. Then a was labelled $\neg u(a)$ by L-U-b: there is a rule r' for a with $\neg u(r')$. Then the label $\neg o(a)$ must be caused by either L-O-c or L-O-d. In both cases, $a \notin \mathcal{Q}$ and there is a rule r' for a that is labelled $\neg u(r')$ and $\neg o(r')$. Since $\texttt{h}(r') \leq k$, we can apply the induction hypothesis: for each $AS' \in F(AS)$, there is an argument based on r in $Arg(AS')$ that is not attacked by an argument in $G(AS')$. Since we chose a arbitrarily, we know that for each $AS' \in F(AS)$, for each $a \in \texttt{ants}(r)$, there is a rule r' for a such that there is an argument based on r' in $Arg(AS')$ that is not attacked by an argument in $G(AS')$. Consider an arbitrary $AS' = (\mathcal{L}, \mathcal{R}, \mathcal{Q}, \mathcal{K}')$ in $F(AS)$. Given $\texttt{ants}(r) = \{a_1, \ldots, a_m\}$, let A_i be the argument for A_i in $Arg(AS')$ that is not attacked by an argument in $G(AS')$. Then by Definition 2, $A : A_1, \ldots, A_m \Rightarrow \texttt{cons}(r)$ is a rule-based argument based on r in $Arg(AS')$. We already concluded that A is not attacked on a subargument by an argument in $G(AS')$. Since $\texttt{cons}(r) \notin \mathcal{Q}$, A is not attacked on its conclusion by an observation-based argument either. By Lemma 4, A is not attacked on its conclusion by an argument in $G(AS')$. So there is an argument based on r in $Arg(AS')$ that is not attacked by an argument in $G(AS')$. Since we chose AS' arbitrary from $F(AS)$, this is the case for every $AS' \in F(AS)$. To conclude: for each $AS' \in F(AS)$, there is an argument based on r in $Arg(AS')$ that is not attacked by any argument in $G(AS')$. □

Lemma 12 (Labelled not defended or blocked)**.** *Given an argumentation setup $AS = (\mathcal{L}, \mathcal{R}, \mathcal{Q}, \mathcal{K})$, let L' be the labelling obtained by* STABILITY *(Algorithm 3). For each $r \in \mathcal{R}$: if r is labelled $\neg d(r)$ and $\neg b(r)$ by L', then for each $AS' \in F(AS)$, each argument based on r in $Arg(AS')$ is attacked by an argument in $G(AS')$.*

Proof. Suppose that r is a rule in $\mathcal{R}$ such that r is labelled $\neg d(r)$ and $\neg b(r)$ by L'.

First suppose that r is labelled unsatisfiable after the preprocessing step: $L_p(r) = \langle 1,0,0,0 \rangle$. This implies that r has not been labelled $\langle 1,1,1,1 \rangle$ in Algorithm 2 line 8. That means that the condition $L(r) = \langle 1,0,0,0 \rangle$ and for each $l \in \texttt{ants}(r) : L(l) \neq \langle 1,0,0,0 \rangle$ (Algorithm 2 line 7) did not apply to r. In other words, there must be an $a \in \texttt{ants}(r)$ such that $L_p(a) = \langle 1,0,0,0 \rangle$. Then by Lemma 8, for each $AS' \in F(AS)$: there is no argument for a in $Arg(AS')$, so there is no rule-based argument based on r in $Arg(AS')$. Then trivially, for each $AS' \in F(AS)$, each argument based on r in $Arg(AS')$ is attacked by an argument in $G(AS')$.

Alternatively, $L_p(r) = \langle 1,1,1,1 \rangle$. So r must be labelled $\neg d(r)$ and $\neg b(r)$ by L'. Let $c(r)$ be the number of the iteration of the while loop (Algorithm 1 line 7–16) in which r is relabelled for the last time and let L_i be the labelling immediately after the i'th iteration of the while loop. We show by induction on $c(r)$ that for each $AS' \in F(AS)$, each argument based on r in $Arg(AS')$ is attacked by an argument in $G(AS')$.

We choose $c(r) = 1$ as base case. If $c(r) = 0$, then r is not relabelled in the while-loop, but rules cannot be relabelled between the preprocessing step and the first iteration of the while-loop. This implies that $L'(r) = L_p(r) = \langle 1,1,1,1 \rangle$, which contradicts our assumption that r is labelled $\neg d(r)$ and $\neg b(r)$ by L'.

Base case: Suppose that $c(r) = 1$. Then r is relabelled for the last time in the first iteration of the while-loop. By Definition 9, the fact that r is labelled $\neg d(r)$ must have been caused by case R-D-a: there is an antecedent a of r that is labelled $\neg d(a)$. Furthermore, r is labelled $\neg b(r)$. This can be caused by either case R-B-a or R-B-b of Definition 9, but in both cases, there is an antecedent a of r that is labelled $\neg d(a)$ and $\neg b(a)$. Let a be the antecedent of r that is labelled $\neg d(a)$ and $\neg b(a)$. We consider two cases: either $a \in \mathcal{Q}$ or $a \notin \mathcal{Q}$.

- First suppose that $a \in \mathcal{Q}$. Then the fact that a is labelled $\neg d(a)$ must be caused by Definition 9 case L-D-a: $-a \in \mathcal{K}$. Then there is an observation-based argument for $-a$ in $Arg(AS)$. By Lemma 5, for each $AS' \in F(AS)$, there is an observation-based argument for $-a$ in $Arg(AS')$. By Lemma 4, for each $AS' \in F(AS)$, each argument for a is attacked by an argument in $G(AS')$.

- Alternatively suppose that $a \notin \mathcal{Q}$. We have that $c(r) = 1$, so r must have been added to TODO-SET by Algorithm 1 line 6 and a must have been relabelled for the last time by line 5. It also means that the condition in line 4 must have been true. Given that $a \notin \mathcal{Q}$, this means that there is no rule for a in $\mathcal{R}$. So for each $AS' \in F(AS)$ there is no observation-based argument for a ($a \notin \mathcal{Q}$) and there is no rule-based argument for a (there is no rule for a in $\mathcal{R}$) in $Arg(AS')$. Then for each $AS' \in F(AS)$,

there is no argument for a in $Arg(AS)$. This implies that for each AS' in $F(AS)$, each argument for a is attacked by an argument in $G(AS')$.

To conclude, in both cases, for each AS' in $F(AS)$, each argument for a is attacked by an argument in $G(AS')$.

Induction hypothesis: For each $r \in \mathcal{R}$ with $c(r) \leq k$, for each $AS' \in F(AS)$, each argument based on $r \in Arg(AS')$ is attacked by an argument in $G(AS')$.

Induction step: Now suppose that $c(r) = k+1$. Consequently, for each antecedent $a \in \texttt{ants}(r)$: $c(a) \leq k$: rules can only be relabelled by Algorithm 1 line 9, so they must have been added to TODO-SET after the labelling of an antecedent changed in a previous iteration of the while loop. We assumed that r is labelled $\neg d(r)$ and $\neg b(r)$. The fact that r is labelled $\neg d(r)$ must have been caused by Definition 9 case R-D-a: there is an antecedent a of r that is labelled $\neg d(a)$. The fact that r is labelled $\neg b(r)$ can be caused by either R-B-a or R-B-b, but in both cases, there is an antecedent a of r that is labelled $\neg d(a)$ and $\neg b(a)$. Next, we distinguish two cases: $a \in \mathcal{Q}$ and $a \notin \mathcal{Q}$.

If $a \in \mathcal{Q}$, then $\neg d(a)$ must have been caused by Definition 9 case L-D-a. This implies that $-a \in \mathcal{K}$. So there is an observation-based argument A for $-a$ in $Arg(AS)$. By Lemma 5, A is in $Arg(AS')$ for every $AS' \in F(AS)$. Being observation-based, A cannot be attacked and therefore $A \in G(AS')$ for every $AS' \in F(AS)$. If there is an argument for a in any $AS' \in F(AS)$, then it is attacked by $A \in G(AS')$.

Now assume that $a \notin \mathcal{Q}$. This implies that each rule r' for a is labelled $\neg d(r') \wedge \neg b(r')$, as we will prove next. Given that $a \notin \mathcal{Q}$, the fact that a is labelled $\neg d(a)$ can be caused by either case L-D-b or L-D-c of Definition 9.

- First suppose that case L-D-c applies. Then there is a rule r'' for $-a$ with $\neg u(r'')$ and $\neg o(r'')$. We also assumed that a is labelled $\neg b(a)$. This cannot be caused by case L-B-a, since $a \notin \mathcal{Q}$. It cannot be caused by case L-B-c or L-B-d either: that would mean that there would be a rule r'' for $-a$ labelled $\langle 0,0,0,0\rangle$, which would contradict Lemma 10. So the fact that a is labelled $\neg b(a)$ must have been caused by case L-B-b: each rule r' for a is labelled $\neg d(r')$ and $\neg b(r')$.

- Alternatively, suppose that a is labelled $\neg d(a)$ by case L-D-b. Then for each rule r' for a: $\neg d(r')$. Again, the fact that a is labelled $\neg b(a)$ cannot be cause by case L-B-a since $a \notin \mathcal{Q}$. It cannot be caused by case L-B-d either: that would imply that there would be a rule r' for a that is labelled $\langle 0,0,0,0\rangle$, which would contradict Lemma 10. So either case L-B-b or L-B-c applies. In both cases, each rule r' for a is labelled $\neg b(r')$. Together with our earlier conclusion that $\neg d(r')$ for each rule r' for a, we derive that each rule r' for a is labelled $\neg d(r')$ and $\neg b(r')$.

To conclude, each rule r' for a is labelled $\neg d(r')$ and $\neg b(r')$. Next, we apply the induction hypothesis for each rule r' for a. $c(a) \leq k$, so $c(r') \leq k$. By the induction hypothesis: for each $AS' \in F(AS)$, for each rule r' for a: each argument based on r' in $Arg(AS')$ is attacked by an argument in $G(AS')$. There are no observation-based arguments for a, since $a \notin \mathcal{Q}$. So for each $AS' \in F(AS)$, each argument for a in $Arg(AS')$ is attacked by an argument in $G(AS')$.

We can conclude that for each $a \in \mathcal{L}$ (if $a \in \mathcal{Q}$ or $a \notin \mathcal{Q}$) for each $AS' \in F(AS)$, each argument for a in $Arg(AS')$ is attacked by an argument in $G(AS')$. Then for each $AS' \in F(AS)$, each argument based on r in $Arg(AS')$ is attacked on its subargument for a by an argument in $G(AS')$. □

Now that we have proved Lemma 9–12, we can use these lemmas to show that the STABILITY algorithm is sound: if STABILITY labels a literal l as stable in an argumentation setup AS, then l is stable in AS. We prove this in Proposition 2.

Proposition 2 (Soundness stability labelling)**.** *Given an argumentation setup $AS = (\mathcal{L}, \mathcal{R}, \mathcal{Q}, \mathcal{K})$ and labelling L' obtained by* STABILITY *(Algorithm 3), if a literal $l \in \mathcal{L}$ is labelled stable in AS, then l is stable in AS.*

Proof. Suppose that l is labelled stable by STABILITY (Algorithm 3) and let L' be the obtained labelling. Then $L'(l) = \langle 1,0,0,0\rangle$, $L'(l) = \langle 0,1,0,0\rangle$, $L'(l) = \langle 0,0,1,0\rangle$ or $L'(l) = \langle 0,0,0,1\rangle$. In Lemma 8, we already showed that l is unsatisfiable in every $AS' \in F(AS)$ if l is labelled $\langle 1,0,0,0\rangle$ by Algorithm 2 (i.e. $L_p(l) = \langle 1,0,0,0\rangle$). Next, we assume that $L_p(l) \neq \langle 1,0,0,0\rangle$. So l must be labelled stable by Algorithm 1.

Let $c(l)$ be the number of the iteration of the while loop (Algorithm 1 line 7–16) in which l is relabelled for the last time and let L_i be the labelling immediately after the i'th iteration of the while loop. We show by induction on $c(l)$ that l is stable in AS.

Base case: First assume that $c(l) = 0$. We assumed that l was not labelled stable in the preprocessing step, so $L_p(l) \neq \langle 1,0,0,0\rangle$; therefore $L_p(l) = \langle 1,1,1,1\rangle$. However, we assumed that l is labelled stable by STABILITY, so l must be relabelled afterwards. Given that $c(l) = 0$, this must have happened in Algorithm 1 line 5.

First suppose that l was labelled $u(l)$ after executing this line. Given our assumption that l is labelled stable, this implies that $L'(l) = \langle 1,0,0,0\rangle$. Remember that $L_p(l) = \langle 1,1,1,1\rangle$. This must have happened in Algorithm 2 line 2 or line 9. However, we will show that in both cases l would be labelled $L'(l) \neq \langle 1,0,0,0\rangle$ by Definition 9.

- If l was labelled $L_p(l) = \langle 1,1,1,1\rangle$ in Algorithm 2 line 2, then $l \in \mathcal{Q}$ and $-l \notin \mathcal{K}$. Then L-D-a, L-D-b and L-D-c do not apply, so l is labelled $d(l)$. This means that $L'(l) \neq \langle 1,0,0,0\rangle$.
- Alternatively, l was labelled $L_p(l) = \langle 1,1,1,1\rangle$ in Algorithm 2 line 9. This means that $l \notin \mathcal{Q} \vee -l \in \mathcal{K}$ and there is a rule r for l that is labelled $L_p(r) = \langle 1,1,1,1\rangle$ just before (in Algorithm 2 line 8). The fact that $l \notin \mathcal{Q} \vee -l \in \mathcal{K}$ implies that $l \notin \mathcal{K}$, so Definition 8 case L-O-a does not apply. Given that there is a rule r for l that is labelled $L_p(r) = \langle 1,1,1,1\rangle$, cases L-O-b and L-O-c do not apply either. Finally, note that rules are not relabelled after the preprocessing step and in the preprocessing step all rules $r \in \mathcal{R}$ are labelled $u(r)$. That means that case L-O-d also cannot apply for l. As a result, l must be labelled $o(l)$, so $L'(l) \neq \langle 1,0,0,0\rangle$.

We derived that in both cases $L'(l) \neq \langle 1,0,0,0\rangle$. This contradicts our earlier conclusion that $L'(l) = \langle 1,0,0,0\rangle$. So we must retract our assumption that l was labelled $u(l)$.

Then l is labelled $\neg u(l)$. Given that all rules $r \in \mathcal{R}$ are labelled $u(r)$, this cannot be caused by Definition 8 case L-U-b. Therefore case L-U-a must have applied: $l \in \mathcal{K}$. Then l is labelled $\neg o(l)$ by case L-O-a and $\neg b(l)$ by case L-B-a, so l is labelled $L'(l) = \langle 0,1,0,0\rangle$ (defended). Next we show that for each $AS' \in F(AS)$, l is defended in AS'. By Definition 7, for each $AS' = (\mathcal{L},\mathcal{R},\mathcal{Q},\mathcal{K}')$ in $F(AS)$, $l \in \mathcal{K}'$. So for each AS' in $F(AS)$, there is an observation-based argument for l in $Arg(AS')$. Since observation-based arguments cannot be attacked (Definition 4, there is an argument for l in the grounded extension $G(AS')$ of each future setup $AS' \in F(AS)$. So for each $AS' \in F(AS)$, l is defended in AS'. This means that for each l such that $c(l) = 0$: if l is labelled stable in AS, then l is stable in AS.

Induction hypothesis: If a literal $l \in \mathcal{L}$ with $c(l) \leq k$ is labelled stable in AS, then l is stable in AS.

Induction step: Now consider a literal $l \in \mathcal{L}$ such that $c(l) = k+1$ and assume that l is labelled stable. Then l is either labelled unsatisfiable, defended, out or blocked. Next, we consider all four cases.

First assume that l is labelled **unsatisfiable**: $L'(l) = \langle 1,0,0,0\rangle$. We need to prove that for each $AS' \in F(AS)$, there is no argument for l in $Arg(AS')$. By Definition 2, we distinguish observation-based and rule-based arguments.

First, we prove that for each $AS' \in F(AS)$, there is no **observation-based** argument for l in $Arg(AS')$. We consider the cases that $l \in \mathcal{Q}$ and $l \notin \mathcal{Q}$. First, suppose that $l \in \mathcal{Q}$. We know that l is labelled $\neg d(l)$ (since it is labelled unsatisfiable). Given that $l \in \mathcal{Q}$, this must have been caused by Definition 9 case L-D-a: $-l \in \mathcal{K}$. Then for each $AS' \in F(AS)$, there is an observation-based argument for $-l$ in $Arg(AS')$. This implies that for each $AS' = (\mathcal{L},\mathcal{R},\mathcal{Q},\mathcal{K}')$ in $F(AS)$, $l \notin \mathcal{K}'$ by Definition 1. Alternatively, $l \notin \mathcal{Q}$. By Definition 1, for each $AS' \in F(AS)$, $l \notin \mathcal{K}'$. To conclude, for each $AS' \in F(AS)$, there is no observation-based argument for l in $Arg(AS')$.

Next, we will prove that for each AS' in $F(AS)$, there is no **rule-based** argument for l in $Arg(AS')$ either. In order to prove this, we will first show that each rule r for l is labelled $L'(r) = \langle 1,0,0,0\rangle$. Again, we consider the cases that $l \in \mathcal{Q}$ and $l \notin \mathcal{Q}$.

- First suppose that $l \in \mathcal{Q}$. We know that $l \notin \mathcal{K}$, so $\neg o(l)$ must have been caused by Definition 9 case L-O-b. Therefore, for each rule r for l: $\neg d(r)$, $\neg o(r)$ and $\neg b(r)$. Then by Lemma 10, each rule r for l is labelled $L'(r) = \langle 1,0,0,0\rangle$.
- Now suppose that $l \notin \mathcal{Q}$. Since l is labelled $u(l)$, we know that $u(r)$ for each rule r for l: otherwise the u-boolean of l would have been labelled False by Definition 9 case L-U-b. This implies that

$\neg b(l)$ cannot have been caused by L-B-d. So it must have been caused by either L-B-b or L-B-c. Next, we consider both cases:

- If $\neg b(l)$ has been caused by L-B-b, then for each rule r for l: $\neg d(r)$ and $\neg b(r)$.
- Otherwise, $\neg b(l)$ was caused by L-B-c, so for each rule r for l is lablled $\neg b(r)$ and for each rule r' for $-l$ is labelled $\neg d(r')$ and $\neg b(r')$. By Lemma 10, this implies that $\neg d(l)$ cannot have been caused by L-D-c, so this must be caused by L-D-b. Then for each rule r for l: $\neg d(r)$ and $\neg b(r)$.

To conclude, in both cases each rule r for l is labelled $\neg d(r)$ and $\neg b(r)$. Then by Lemma 10, $\neg o(l)$ cannot have been caused by L-O-d, so it must have been caused by either L-O-b or L-O-c. In both cases, for each rule r for l: $\neg d(r)$, $\neg o(r)$ and $\neg b(r)$. By Lemma 10, this implies that each rule r for l is labelled $\langle 1,0,0,0\rangle$.

We conclude that for each $l \in \mathcal{L}$ with $c(l) = k+1$, each rule r for l is labelled $\langle 1,0,0,0\rangle$.

Now let r be an arbitrary rule r for l in $\mathcal{R}$. From our earlier conclusion, we know that this rule (like any rule for l) must be labelled $\neg d(r)$, $\neg o(r)$ and $\neg b(r)$. Then by Definition 9 case R-D-a and $\neg d(r)$, there is an antecedent $a \in \texttt{ants}(r)$ that is labelled $\neg d(a)$. The fact that r is labelled $\neg b(r)$ can be caused by either case R-B-a or R-B-b, but in both cases, there is an antecedent $a \in \texttt{ants}(r)$ that is labelled $\neg d(a)$ and $\neg b(a)$. Similarly, by case R-O-a or R-O-b, we derive that there is an antecedent $a \in \texttt{ants}(r)$ that is labelled $\neg d(a)$, $\neg o(a)$ and $\neg b(a)$.

By Lemma 10, this implies that there is an antecedent a such that $L'(a) = \langle 1,0,0,0\rangle$. $c(a) \leq k$, so by the induction hypothesis, for each $AS' \in F(AS)$, a is unsatisfiable in AS'. So for each $AS' \in F(AS)$, there is no argument for a in $Arg(AS')$. Then by Definition 2, for each AS' in $F(AS)$ there is no rule-based argument for l based on r in $Arg(AS')$. Since we chose r arbitrary, this is the case for every rule r for l, we can derive that for each $AS' \in F(AS)$, there is no rule-based argument for l in $Arg(AS')$.

Given our earlier conclusion that for each $AS' \in F(AS)$, there is no observation-based argument for l in $Arg(AS')$, we conclude that for each $AS' \in F(AS)$, there is no argument for l in $Arg(AS')$. By Definition 6, for each $AS' \in F(AS)$ l is unsatisfiable in AS'.

Now assume that l is labelled **defended**: $L'(l) = \langle 0,1,0,0\rangle$. We distinguish two cases: either $l \in \mathcal{Q}$ or $l \notin \mathcal{Q}$.

- First suppose that $l \in \mathcal{Q}$. We will show by contradiction that $l \in \mathcal{K}$: suppose that $l \notin \mathcal{K}$. Then $\neg u(l)$ must have been caused by Definition 9 case L-U-b and $\neg o(l)$ must have been caused by case L-O-b. So there is a rule r for l with $\neg u(r)$ (L-U-b) and for each rule r for l: $\neg d(r)$, $\neg o(r)$ and $\neg b(r)$ (L-O-b). But this implies that there exists a rule r for l such that $\neg u(r)$, $\neg d(r)$, $\neg o(r)$ and $\neg b(r)$, which contradicts Lemma 10. So $l \in \mathcal{K}$. By Definition 7, for each $AS' = (\mathcal{L},\mathcal{R},\mathcal{Q},\mathcal{K}')$ in $F(AS)$, $l \in \mathcal{K}'$. So for each AS' in $F(AS)$, there is an observation-based argument for l in $Arg(AS')$. Since observation-based arguments cannot be attacked (Definition 4), there is an argument for l in the grounded extension $G(AS')$ of each future setup $AS' \in F(AS)$. So for each $AS' \in F(AS)$, l is defended in AS'.

- Next, we consider the alternative case that $l \notin \mathcal{Q}$. Then $\neg u(l)$ must have been caused by Definition 9 case L-U-b: there is a rule r for l with $\neg u(r)$. By Lemma 10, the fact that $\neg o(l)$ cannot have been caused by L-O-b. Since we also know that $l \notin \mathcal{K}$, $\neg o(l)$ must have been caused by either L-O-c or L-O-d. In both cases, there is a rule r for l with $\neg u(r)$ and $\neg o(r)$. This implies that $\neg b(l)$ must have been caused by either L-B-c or L-B-d. In both cases, there is a rule r for l with $\neg u(r)$, $\neg o(r)$ and $\neg b(r)$ and for each rule r' for $-l$: $\neg d(r')$ and $\neg b(r')$.

  Now let r be the rule for l such that $\neg u(r)$, $\neg o(r)$ and $\neg b(r)$. The fact that r is labelled $\neg u(r)$ must have been caused by R-U-a: for each antecedent a of r: $\neg u(a)$. Then $\neg o(r)$ has to be caused by R-O-a (R-O-b would contradict Lemma 10) so for each antecedent a of r: $\neg u(a)$ and $\neg o(a)$. Furthermore, $\neg b(r)$ must have been caused by R-B-a (again, R-B-b would contradict Lemma 10), so each antecedent a of r is labelled $\neg u(a)$, $\neg o(a)$ and $\neg b(a)$. By Lemma 10, $L'(a) = \langle 0,1,0,0\rangle$. $c(l) = k+1$, so $c(a) \leq k$; therefore we can apply the induction hypothesis: for each antecedent a of r,

for each $AS' \in F(AS)$, a is defended in AS'. So by Definition 8, for each $a \in \texttt{ants}(r)$, for each AS' in $F(AS)$, there is an argument for a in $G(AS')$. By Lemma 3, each argument attacking this argument is attacked by an observation-based argument. To conclude, for each $AS' \in F(AS)$, for each argument A based on r in $Arg(AS')$, each argument B attacking A *on a subargument* in AS' is attacked by an observation-based argument in $Arg(AS')$.

Now we show that for each $AS' \in F(AS)$, for each argument A based on r in $Arg(AS')$, each argument B attacking A *on its conclusion* in AS' is attacked by an observation-based argument in $Arg(AS')$ as well. For this, we use our earlier conclusion that each rule r' for $-l$ is labelled $\neg d(r')$ and $\neg b(r')$. By Lemma 12, this implies that for each $AS' \in F(AS)$, for each rule r' for $-l$, each argument based on r in $Arg(AS')$ is attacked by an argument in $G(AS')$. So for each $AS' \in F(AS)$, there is an argument A based on r such that each argument B attacking A is attacked by an observation-based argument in $Arg(AS')$. Then by Lemma 3, for each AS' in $F(AS)$, $A \in G(AS')$. In other words, by Definition 6, for each AS' in $F(AS)$, l is defended in AS'.

To conclude, if $L'(l) = \langle 0,1,0,0 \rangle$, then in all cases l is stable in AS (Definition 8).

Alternatively, assume that l is labelled **out**: $L'(l) = \langle 0,0,1,0 \rangle$. We consider both the case that $l \in \mathcal{Q}$ and the case that $l \notin \mathcal{Q}$.

- First suppose that $l \in \mathcal{Q}$. Then $\neg d(l)$ must have been caused by Definition 9 L-D-a, so $-l \in \mathcal{K}$. Knowledge bases are consistent by Definition 1, therefore $l \notin \mathcal{K}$. This means that $\neg u(l)$ must have been caused by Definition 9 case L-U-b: there is a rule r for l with $\neg u(r)$. Then by Lemma 9, there is a rule-based argument for l based on r in $Arg(AS)$. By definition of attack (Definition 4), there is an argument for l in $Arg(AS)$ but each argument for l is attacked by the observation-based argument $-l$. Then by Lemma 5: for each $AS' \in F(AS)$: there is a rule-based argument for l in $Arg(AS')$ which is attacked by the observation $-l$ in $Arg(AS')$. Finally, by Lemma 4, for each $AS' \in F(AS)$, l is out in AS'.

- Now we consider the alternative case that $l \notin \mathcal{Q}$. Then $l \notin \mathcal{K}$ (Definition 1) so $\neg u(l)$ must have been caused by Definition 9 case L-U-b: there is a rule r for l with $\neg u(r)$. Then by Lemma 9, there is a rule-based argument based on r in $Arg(AS)$. Furthermore, since l is labelled $o(l)$, we know that there is no rule r for l with $\neg u(r)$ and $\neg o(r)$ (otherwise Definition 9 case L-O-d would apply). As a result, L-B-c cannot apply. So the fact that l is labelled $\neg b(l)$ must be caused by either L-B-b or L-B-c. In both cases, for each rule r for l: $\neg d(r)$ and $\neg b(r)$, as we will show next. For the case that L-B-b applies, this is immediately clear. If $\neg b(l)$ was caused by L-B-c, then for each rule r for l: $\neg b(r)$ and for each rule r' for $-l$: $\neg d(r')$ and $\neg b(r')$. Then, by Lemma 10, the fact that $\neg d(l)$ cannot have been caused by L-D-c, so must have been caused by L-D-b: for each rule r for l: $\neg d(r)$. To conclude, for each rule r for l: $\neg d(r)$ and $\neg b(r)$.

  Now we can apply Lemma 12: for each rule r for l, for each AS' in $F(AS)$, each argument based on r in $Arg(AS')$ is attacked by an argument in $G(AS')$. So for each AS' in $F(AS)$, each rule-based argument in $Arg(AS')$ is attacked by an argument in $G(AS')$. Together with our earlier conclusion that there is a rule-based argument for l in $Arg(AS')$ for each $AS' \in F(AS)$, by Definition 6, l is out in AS' for each AS in $F(AS)$.

So by Definition 8, l is stable in AS.

Finally, assume that l is labelled **blocked**: $L'(l) = \langle 0,0,0,1 \rangle$.

This implies that $l \notin \mathcal{Q}$ (Definition 9 case L-B-a). Then also $l \notin \mathcal{K}$, so $\neg u(l)$ must be caused by case L-U-b: there is a rule r for l with $\neg u(r)$. $\neg o(l)$ can be caused by either L-O-c or L-O-d. In both cases, there is rule r for l with $\neg u(r)$ and $\neg o(r)$. By Lemma 11, for each $AS' \in F(AS)$, there is an argument based on r in $Arg(AS')$ that is not attacked by an argument in $G(AS')$.

In order to prove that l is blocked in each $AS' \in F(AS)$, we now only need to prove that for each $AS' \in F(AS)$, there is no argument for l in $G(AS')$. For this purpose, we use the fact that l is labelled $\neg d(l)$ by L'. This implies that for every future argumentation setup AS' in $F(AS)$ there is no argument for l in the grounded extension $G(AS')$, as we will show next by induction.

Consider a literal a such that a is labelled $\neg d(a)$ by L'.

Base case: First suppose that $c(a) = 0$. Then either $-a \in \mathcal{K}$ (by Definition 9 L-D-a) or $a \notin \mathcal{Q}$ (by L-D-b or L-D-c). In all cases, there is no $\mathcal{K}'$ such that $a \in \mathcal{K}'$, so there is no observation-based argument for a in any future setup. Also, there is no rule for a ($c(a) = 0$), so there is no rule-based argument for a either in any future setup. Given that there is no argument for a in $Arg(AS')$, there also cannot be an argument for a in $G(AS')$ for any AS' in $F(AS)$.

Induction hypothesis: If a literal $a \in \mathcal{L}$ with $c(a) \leq k$ is labelled $\neg d(a)$, then for every future argumentation setup AS' in $F(AS)$ there is no argument for a in the grounded extension $G(AS')$.

Induction step: Now suppose that $c(a) = k+1$. By Definition 9, $\neg d(a)$ can be caused by three cases: L-D-a, L-D-b or L-D-c.

- If $\neg d(a)$ was caused by L-D-a, then $-a \in \mathcal{K}$. So there is an observation-based argument for $-a$ in $Arg(AS)$. By Lemma 5, this argument is also in $Arg(AS')$ for every future AS' in $F(AS)$. By Definition 4, observation-based arguments cannot be attacked, so by Definition 5, the observation-based argument for $-a$ is in the grounded extension of every AS'. So for every AS': if there exists an argument for l in $Arg(AS')$, then it must be rule-based and is attacked by the observation-based argument $-a$, which is in the grounded extension. Therefore there is no argument for a in $G(AS')$ for any AS' in $F(AS)$.

- If $\neg d(a)$ was caused by L-D-b, then $a \notin \mathcal{Q}$ and for each rule r for a: $\neg d(r)$. $a \notin \mathcal{Q}$ implies that there is no observation-based argument for a in $Arg(AS')$ for any $AS' \in F(AS)$. $c(a) = k+1$, so there exists at least one rule for a. We take an arbitrary rule r for a. The fact that $\neg d(r)$ for every rule r for a implies that for every rule r for a, there is an a_i in $\texttt{ants}(r)$ with $\neg d(a_i)$ (R-D-a). $c(a_i) \leq k$, so by the induction hypothesis, there is no argument for a_i in the grounded extension of any future argumentation setup AS'. By Lemma 3, each argument for a_i in each $Arg(AS')$ in each $AS' \in G(AS)$ is attacked and not each attacker of a_i is attacked by an observation-based argument. Then every rule-based argument A for a based on r in each $Arg(AS')$ in each $AS' \in F(AS)$ is attacked and not each attacker of A is attacked by an observation-based argument. Since we chose r arbitrarily, we know that there is no argument for a in $G(AS')$ for any AS' in $F(AS)$.

- Otherwise, $\neg d(a)$ was caused by L-D-c. Then $a \notin \mathcal{Q}$ and there is a rule r' for $-a$ with $\neg u(r') \wedge \neg o(r')$. By Lemma 11, for each future setup AS' in $F(AS)$ there exists an argument for $-a$ based on r' that is not attacked by any argument in $G(AS')$. So for every future setup AS', if an argument for a exists, then it cannot be in $G(AS')$.

To conclude, for each $a \in \mathcal{L}$ such that a is labelled $\neg d(a)$: for each $AS' \in F(AS)$, there is no argument for a in $G(AS')$.

Given that l is labelled $\neg d(l)$, we have that for each $AS' \in F(AS)$, there is no argument for l in $G(AS')$. In combination with our earlier conclusion that for each $AS' \in F(AS)$ there is an argument for l in $Arg(AS')$ that is not attacked by an argument in $G(AS')$, we have that for each $AS' \in F(AS)$, l is blocked in AS' (Definition 6), so l is stable in AS (Definition 8).

We have now shown for each of the four cases of stability as defined in Definition 8: if l is labelled stable in AS, then l is stable in AS. In other words: STABILITY is sound.

□

3.4.2 Conditional completeness

Next, we consider the completeness of our algorithm. As we illustrate in Example 10, STABILITY is not complete for all argumentation setups.

Example 10 (Example 4 continued)**.** We return to our running example on fraud. Consider the argumentation setup $AS = (\mathcal{L}, \mathcal{R}, \mathcal{Q}, \mathcal{K})$ where $\mathcal{L}$, $\mathcal{R}$ and $\mathcal{Q}$ correspond to the language, rules and queryables in Figure 1, but $\mathcal{K} = \{\neg sm, rm\}$. STABILITY does not label f stable: it expects a future argument for f based on $sd, \neg rd, d \Rightarrow f$, where the argument for sd is based on $sp, \neg b \Rightarrow sd$ and the argument for $\neg rd$ is based on $\neg rp, b \Rightarrow rd$. However, this argument would require both b and $\neg b$ to be in the knowledge base, which

violates the consistency criterion. In fact, for each AS' in $F(AS)$ there is no argument for f in $Arg(AS')$, so f should be labelled $\langle 1,0,0,0\rangle$.

Example 10 shows that there are argumentation setups where the STABILITY algorithm wrongfully takes the possibility into account that there exists an argument for a literal in a future argumentation setup. Specifically, this issue is caused by an inconsistency in *potential arguments*, which we define next.

Definition 10 (Potential argument)**.** Let $AS = (\mathcal{L}, \mathcal{R}, \mathcal{Q}, \mathcal{K})$ be an argumentation setup. A **potential argument** A^p inferred from AS is:

- an **observation-based potential argument** c iff $c \in \mathcal{Q}$ and $-c \notin \mathcal{K}$.
  $\texttt{prem}(A^p) = \{c\}$;
  $\texttt{conc}(A^p) = c$;
  $\texttt{sub}(A^p) = \{c\}$;
  $\texttt{def-rules}(A^p) = \emptyset$;
  $\texttt{h}(A^p) = 0$;
  $\texttt{dirsub}(A^p) = \emptyset$.

- a **rule-based potential argument** $A_1, \ldots, A_m \Rightarrow c$ iff there is a rule $c_1, \ldots, c_m \Rightarrow c$ in $\mathcal{R}$ and for each $i \in [1 \,..\, m]$: A_i is a potential argument inferred from AS and $\texttt{conc}(A_i) = c_i$.
  $\texttt{prem}(A^p) = \texttt{prem}(A_1) \cup \ldots \cup \texttt{prem}(A_m)$;
  $\texttt{conc}(A^p) = c$;
  $\texttt{sub}(A^p) = \texttt{sub}(A_1) \cup \ldots \cup \texttt{sub}(A_m) \cup \{A\}$;
  $\texttt{def-rules}(A^p) = \texttt{def-rules}(A_1) \cup \ldots \cup \texttt{def-rules}(A_m)$;
  $\texttt{h}(A^p) = 1 + \max(\texttt{h}(A_1), \ldots, \texttt{h}(A_m))$;
  $\texttt{dirsub}(A^p) = \{A_1, \ldots, A_m\}$.

We refer to a potential argument with conclusion c as "a potential argument for c".
Given some $A^p, B^p \in P(AS)$, A^p is **inconsistent** with B^p iff $\{a, -a\} \in \texttt{prem}(A^p) \cup \texttt{prem}(B^p)$ for some $a \in \mathcal{L}$.
For two potential arguments $A^p, B^p \in P(AS)$ we say that A^p **p-attacks** B^p iff A^p's conclusion is c and either:

- **attack on conclusion**: $-c$ is the conclusion of B^p and $-c \notin \mathcal{K}$.
- **attack on subargument**: $-c$ is the conclusion of a subargument B' of B^p such that $B' \neq B^p$ and $-c \notin \mathcal{K}$.

We write that "A^p p-attacks B^p on B'" if A^p attacks B^p, $B' \in \texttt{sub}(B^p)$ and $\texttt{conc}(A^p) = -\texttt{conc}(B')$.

Just like arguments, potential arguments may be circular. Circularity of potential arguments is defined next.

Definition 11 (Circularity of potential arguments)**.** Let $AS = (\mathcal{L}, \mathcal{R}, \mathcal{Q}, \mathcal{K})$ be an argumentation setup and let A^p be a potential argument in $P(AS)$. A^p is circular iff there is a sequence $\langle A_1, \ldots, A_m\rangle$ such that $A_m \in \texttt{sub}(A^p)$, $\texttt{conc}(A_m) = \texttt{conc}(A_1)$ and for each $i \in [1 \,..\, m-1]$: $A_i \in \texttt{dirsub}(A_{i+1})$ and $A_i \neq A_{i+1}$. A^p is non-circular iff A^p is not circular.

Analogous to Lemmas 1 and 2, we have the following two lemmas:

Lemma 13 (Existence of non-circular potential arguments)**.** *Let $AS = (\mathcal{L}, \mathcal{R}, \mathcal{Q}, \mathcal{K})$ be an argumentation setup and literal $l \in \mathcal{L}$. There is a potential argument for l in $P(AS)$ iff there is a non-circular potential argument for l in $P(AS)$.*

Lemma 14 (Finite height and non-circularity of potential arguments)**.** *Given a potential argument A^p, if A^p is non-circular, then $\mathfrak{h}(A^p)$ is finite.*

Next, we prove that PREPROCESS (Algorithm 2) labels literals/rules as $\langle 1,0,0,0\rangle$ if and only if there exists no potential argument for/based on them.

Lemma 15 (Soundness and completeness Algorithm 2 for potential arguments)**.** *Given an argumentation setup $AS = (\mathcal{L}, \mathcal{R}, \mathcal{Q}, \mathcal{K})$ and labelling L_p after executing Algorithm 2. For each rule $r \in \mathcal{R}$: $L_p(r) = \langle 1,0,0,0\rangle$ iff there is no $A^p \in P(AS)$ such that A^p is based on r. For each literal $l \in \mathcal{L}$: $L_p(l) = \langle 1,0,0,0\rangle$ iff there is no $A^p \in P(AS)$ such that A^p is a potential argument for l.*

Proof. We will show that Algorithm 2 is sound and complete for potential arguments, that is: for each rule $r \in \mathcal{R}$: $L_p(r) = \langle 1,0,0,0\rangle$ iff there is no $A^p \in P(AS)$ such that A^p is based on r; and for each literal $l \in \mathcal{L}$: $L_p(l) = \langle 1,0,0,0\rangle$ iff there is no $A^p \in P(AS)$ such that A^p is a potential argument for l. We will do this using a loop invariant proof. Note that this approach is different from the induction proof in Lemma 8: that was a soundness proof, whereas we prove both soundness and completeness in the current lemma.

First, we introduce some notation. Let $x \in \mathcal{L} \cup \mathcal{R}$ be a literal or rule and let $L_0^p(x)$ be the label given to x by the preprocessing algorithm (Algorithm 2) between line 4 and 5. Let $L_k^p(x)$ be the label given to x by the preprocessing algorithm just after the k'th iteration of the while loop (line 5–9).

Loop invariant: At iteration i of the while loop, each $l \in \mathcal{L}$ such that there is an A^p for l in $P(AS)$ with $\mathtt{h}(A^p) \leq i$ is labelled $L_i^p(l) = \langle 1,1,1,1\rangle$; otherwise, $L_i^p(l) = \langle 1,0,0,0\rangle$. Furthermore, each $r \in \mathcal{R}$ such that there is an A^p based on r in $P(AS)$ with $\mathtt{h}(A^p) \leq i$ is labelled $L_i^p(r) = \langle 1,1,1,1\rangle$; otherwise, $L_i^p(r) = \langle 1,0,0,0\rangle$.

Initialisation: For each literal $l \in \mathcal{L}$, there is an A^p for l in $P(AS)$ with $\mathtt{h}(A^p) = 0$ iff $l \in \mathcal{Q}$ and $-l \notin \mathcal{K}$ (Definition 10). By Algorithm 2 line 2, $L_0^p(l) = \langle 1,1,1,1\rangle$ iff $l \in \mathcal{Q}$ and $-l \notin \mathcal{K}$ and $L_0^p(l) = \langle 1,0,0,0\rangle$ otherwise. So $L_0^p(l) = \langle 1,0,0,0\rangle$ iff there is no $A^p \in P(AS)$ with $\mathtt{h}(A^p) = 0$ such that A^p is a potential argument for l.

For each rule $r \in \mathcal{R}$, there is no A^p based on r in $P(AS)$ with $\mathtt{h}(A^p) = 0$ (since rule-based arguments have a height of at least 1). By Algorithm 2 line 4, all rules $r \in \mathcal{R}$ are labelled $L_0^p(r) = \langle 1,0,0,0\rangle$. Therefore, for each rule $r \in \mathcal{R}$, $L_0^p(r) = \langle 1,0,0,0\rangle$ iff there is no $A^p \in P(AS)$ with $\mathtt{h}(A^p) = 0$ such that A^p is a potential argument for l.

Maintenance: Assume that the loop invariant holds at the start of iteration i: At iteration i of the while loop, each $l \in \mathcal{L}$ such that there is an A^p for l in $P(AS)$ with $\mathtt{h}(A^p) \leq i$ is labelled $L_i^p(l) = \langle 1,1,1,1\rangle$; otherwise, $L_i^p(l) = \langle 1,0,0,0\rangle$. Furthermore, each $r \in \mathcal{R}$ such that there is an A^p based on r in $P(AS)$ with $\mathtt{h}(A^p) \leq i$ is labelled $L_i^p(r) = \langle 1,1,1,1\rangle$; otherwise, $L_i^p(r) = \langle 1,0,0,0\rangle$.

Now consider a $l \in \mathcal{L}$ such that there is no A^p for l in $P(AS)$ with $\mathtt{h}(A^p) \leq i$ but there is an A^p for l in $P(AS)$ with $\mathtt{h}(A^p) = i+1$. So there is a rule $r \in \mathcal{R}$ for l such that for each $a \in \mathtt{ants}(r)$ there is a potential argument $A^{p\prime}$ for a with $\mathtt{h}(A^{p\prime}) \leq i$. Then by the loop invariant, for each $a \in \mathtt{ants}(r)$: $L_i^p(a) = \langle 1,1,1,1\rangle$. If a literal or rule $x \in \mathcal{L} \cup \mathcal{R}$ is labelled $L_i^p(x) = \langle 1,1,1,1\rangle$ then for each $j \geq i$: $L_j^p(x) = \langle 1,1,1,1\rangle$, so for each $a \in \mathtt{ants}(r)$: $L_{i+1}^p(a) = \langle 1,1,1,1\rangle$. Then $L_{i+1}^p(r) = \langle 1,1,1,1\rangle$ (by Line 8) and $L_{i+1}^p(l) = \langle 1,1,1,1\rangle$ (by Line 9).

Alternatively, consider a $l \in \mathcal{L}$ such that there is no A^p for l in $P(AS)$ with $\mathtt{h}(A^p) \leq i+1$. Then for each rule $r \in \mathcal{R}$ for l, there is an antecedent a such that there is no A^p for a in $P(AS)$ with $\mathtt{h}(A^p) \leq i$. So by the loop invariant, $L_i^p(a) = \langle 1,0,0,0\rangle$. Then no rule r for l, nor l itself, is labelled $\langle 1,1,1,1\rangle$ by Line 8/9. So for each rule r for l: $L_{i+1}^p(r) = \langle 1,0,0,0\rangle$ and for l: $L_{i+1}^p(l) = \langle 1,0,0,0\rangle$.

Termination: From Lemma 7 we know that Algorithm 2 terminates (after polynomial runtime). So there is some finite k such that k is the last iteration of the while loop and at iteration k of the while loop, each $l \in \mathcal{L}$ such that there is an A^p for l in $P(AS)$ with $\mathtt{h}(A^p) \leq k$ is labelled $L_i^p(l) = \langle 1,1,1,1\rangle$; otherwise, $L_k^p(l) = \langle 1,0,0,0\rangle$. There might be an A^p for l/based on r with $\mathtt{h}(A^p) > k$, but then this A^p would be circular. By Lemma 13 there is a non-circular $A^{p\prime}$ for l/based on r with finite $\mathtt{h}(A^{p\prime})$. So then still $L_k^p(l/r) = \langle 1,1,1,1\rangle$ and otherwise $L_k^p(l/r) = \langle 1,0,0,0\rangle$. □

Note that, for a given argumentation setup AS, each argument inferred from AS or some future setup is a potential argument in $P(AS)$. This means that we could also have proved Lemma 8 by using Lemma 15: if a literal l is labelled $L_p(l) = \langle 1,0,0,0\rangle$, then by Lemma 15 there is no potential argument for l in $P(AS)$. This implies that for each $AS' \in F(AS)$, there is no argument for l in $Arg(AS')$. In other words, l is unsatisfiable in AS.

On the other hand, there may be a potential argument $A^p \in P(AS)$ such that there is no $AS' \in F(AS)$ with $A^p \in Arg(AS')$, but then A^p must be inconsistent *with itself*; we encountered this situation in Example 10.

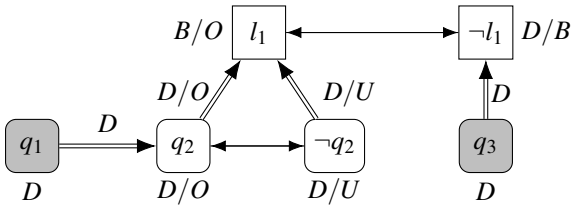

Figure 6: For each $AS' \in F(AS)$: l_1 and $\neg l_1$ are blocked in AS', but not labelled as such due to inconsistent support (l_1) and attack ($\neg l_1$).

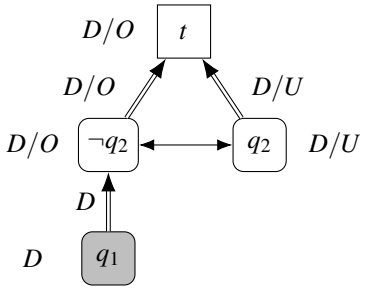

Figure 7: t is inconsistently supported in AS. As a result, t is labelled $u(t)$ or $o(t)$ while for each $AS' = (\mathcal{L}, \mathcal{R}, \mathcal{Q}, \mathcal{K}')$ in $F(AS)$, t is not unsatisfiable or out in AS': if $q_2 \in \mathcal{K}'$ then there is an unattacked argument for t based on $q_2 \Rightarrow t$; otherwise there is an unattacked argument for t based on $\neg q_2 \Rightarrow t$.

The next example reveals another issue, where stability is not detected due to an inconsistency of two *different* potential arguments.

Example 11 (Mutual inconsistency issues)**.** Given the argumentation setup AS illustrated in Figure 6, for each $AS' = (\mathcal{L}, \mathcal{R}, \mathcal{Q}, \mathcal{K}')$ in $F(AS)$, l_1 is blocked in AS': if $\neg q_2 \notin \mathcal{K}'$ then there is an argument for l_1 based on $q_2 \Rightarrow l_1$; otherwise there is an argument for l_1 based on $\neg q_2 \Rightarrow l_1$. However, l_1 is not labelled $\langle 0,0,0,1 \rangle$ because STABILITY wrongfully anticipates a future setup AS' in which each argument for l_1 is attacked by an argument in $G(AS')$ (thus $o(l_1)$). For the same reason, $\neg l_1$ is labelled $d(\neg l_1)$ and therefore $L'(\neg l_1) \neq \langle 0,0,0,1 \rangle$, although $\neg l_1$ is blocked in each $AS' \in F(AS)$.

The issues illustrated in Examples 10 and 11 can be generalised to two situations of inconsistent support or inconsistent attack, as defined next:

Definition 12 (Inconsistently supported/attacked)**.** Given an argumentation setup $AS = (\mathcal{L}, \mathcal{R}, \mathcal{Q}, \mathcal{K})$ and a literal $l \in \mathcal{L}$:

- l is **inconsistently supported** in AS iff there are $A^p, B^p \in P(AS)$ such that $\texttt{conc}(A^p) = \texttt{conc}(B^p) = l$ and A^p is inconsistent with B^p.
- l is **inconsistently attacked** in AS iff there is a potential argument C^p for l in $P(AS)$ and there are $A^p, B^p \in P(AS)$ such that A^p p-attacks C^p, B^p p-attacks C^p and A^p is inconsistent with B^p.

If a potential argument is inconsistent with itself, its conclusion l can be incorrectly labelled $d(l)$ or $b(l)$ (e.g. f in Example 10). Similarly, if two potential arguments with the same conclusion are inconsistent, their conclusion l can be incorrectly labelled $o(l)$ (e.g. l_1 in Example 11). Moreover, if l is inconsistently attacked, l may be incorrectly labelled $d(l)$ (e.g. $\neg l_1$ in Example 11) or $b(l)$. We give additional examples in Example 12–15.

Example 12 (Incorrect labelling $u(l)$ or $o(l)$ due to inconsistent support)**.** Figure 7 shows an example of an argumentation setup AS that is not labelled correctly due to inconsistent support.

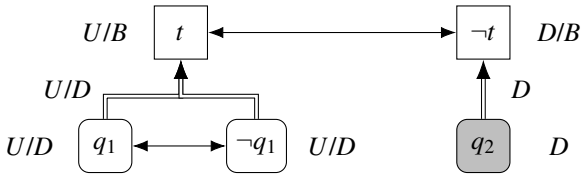

Figure 8: t is inconsistently supported in AS. As a result, t is labelled $b(t)$ while for each $AS' = (\mathcal{L}, \mathcal{R}, \mathcal{Q}, \mathcal{K}')$ in $F(AS)$, t is not blocked in AS': the only potential argument for t is $q_1, \neg q_2 \Rightarrow t$, but this is not an argument in any future argumentation setup.

In every $AS' = (\mathcal{L}, \mathcal{R}, \mathcal{Q}, \mathcal{K}')$ in $F(AS)$, t is not unsatisfiable or out in AS': if $q_2 \in \mathcal{K}'$ then there is an unattacked argument for t based on $q_2 \Rightarrow t$; otherwise there is an unattacked argument for t based on $\neg q_2 \Rightarrow t$. So in this example, for each $AS' \in F(AS)$, t is defended in AS'.

However, t is not labelled $\langle 0,1,0,0 \rangle$ by L'. $q_1 \in \mathcal{K}$, so q_1 is labelled $L'(q_1) = \langle 0,1,0,0 \rangle$ by Definition 9 case L-U-a, L-O-a and L-B-a. $L'(q_1) = \langle 0,1,0,0 \rangle$ by case R-U-a, R-O-a and R-B-a. q_2 and $\neg q_2$ are observable but neither of them is observed. $L'(\neg q_2) = \langle 0,1,1,0 \rangle$ by case L-U-b and L-B-a; $L'(q_2) = \langle 1,1,0,0 \rangle$ by case L-O-b and L-B-a. As a result, $L'(\neg q_2 \Rightarrow t) = \langle 0,1,1,0 \rangle$ by R-U-a and R-B-a; $L'(q_2 \Rightarrow t) = \langle 1,1,0,0 \rangle$ by case R-O-a and R-B-a. Finally, $L'(t) = \langle 0,1,1,0 \rangle$ by case L-U-b and L-B-a.

This issue is caused by the fact that STABILITY wrongfully anticipates a future setup in which each argument for t is attacked by an observation-based argument. But this is impossible: for each $AS' \in F(AS)$, either there is an unattacked argument based on $q_2 \Rightarrow t$ in $Arg(AS')$ or there is an unattacked argument based on $\neg q_2 \Rightarrow t$ in $Arg(AS')$. This issue only occurs when there are two potential arguments for t in $P(AS)$ having mutually inconsistent premises, i.e. if t is inconsistently supported in AS.

Example 13 (Incorrect labelling $d(l)$ or $b(l)$ due to inconsistent support)**.** Figure 8 shows an argumentation setup AS that is not labelled correctly due to inconsistent support. In every $AS' = (\mathcal{L}, \mathcal{R}, \mathcal{Q}, \mathcal{K}')$ in $F(AS)$, t is not defended or blocked in AS', since t is unsatisfiable in AS'. However, t is not labelled $\langle 1,0,0,0 \rangle$ by L', since there is a potential argument based on $q_1, \neg q_1 \Rightarrow t$ in $P(AS)$ and STABILITY does not detect the fact that this potential argument is not an argument in any future argumentation setup $AS' \in F(AS)$.

Example 14 (Incorrect labelling $b(l)$ due to inconsistent attack)**.** Figure 8 shows an argumentation setup AS that is not labelled correctly due to inconsistent support. In every $AS' = (\mathcal{L}, \mathcal{R}, \mathcal{Q}, \mathcal{K}')$ in $F(AS)$, $\neg t$ is not blocked in AS', since t is defended in AS'. However, $\neg t$ is not labelled $\langle 0,1,0,0 \rangle$ by L', since there is a potential argument based on $q_1, \neg q_1 \Rightarrow t$ in $P(AS)$ and STABILITY does not detect the fact that this potential argument is not an argument in any future argumentation setup $AS' \in F(AS)$.

Example 15 (Incorrect labelling $d(l)$ due to inconsistent attack)**.** Figure 9 shows an argumentation setup AS that is not labelled correctly due to inconsistent support. In every $AS' = (\mathcal{L}, \mathcal{R}, \mathcal{Q}, \mathcal{K}')$ in $F(AS)$, t is not defended in AS', since t is blocked in AS'. However, t is not labelled $\langle 0,0,0,1 \rangle$ by L', since two mutually inconsistent potential arguments p-attack the only argument for t.

In the remainder of this section, we will show that, given an argumentation setup $AS = (\mathcal{L}, \mathcal{R}, \mathcal{Q}, \mathcal{K})$ and a literal $l \in \mathcal{L}$, the situations where l is inconsistently supported or attacked in AS are the **only** situations in which L' does not detect l's stability. In order to prove this, we first need Lemma 16, which shows in which situations STABILITY turns one or more booleans (u, d, o and/or b) of the labelling L' to False. This will help us in Proposition 3 to prove in which situations STABILITY turns exactly three booleans of the labelling L' to false. This way we can show in which situations STABILITY is complete.

Lemma 16 (Conditions for labellings)**.** *Given an argumentation setup $AS = (\mathcal{L}, \mathcal{R}, \mathcal{Q}, \mathcal{K})$, let L' be the labelling function from Definition 9 and let $l \in \mathcal{L}$ be a literal. Then:*

1. *If there is an argument for l in $Arg(AS)$, then $\neg u(l)$.*
2. *If each potential argument for l in $P(AS)$ is p-attacked by an observation-based argument in $Arg(AS)$, then $\neg d(l)$ and $\neg b(l)$.*

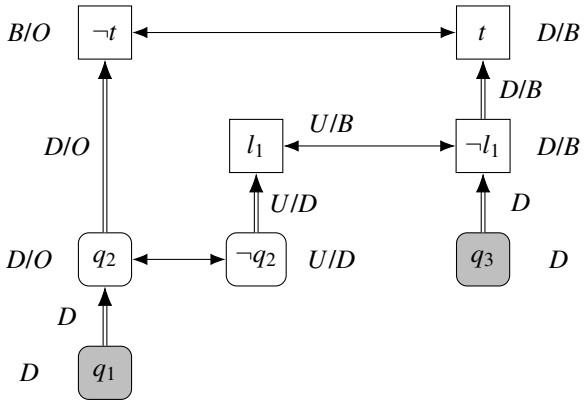

Figure 9: t is inconsistently attacked in AS. As a result, t is labelled $d(t)$ while for each $AS' = (\mathcal{L}, \mathcal{R}, \mathcal{Q}, \mathcal{K}')$ in $F(AS)$, t is not defended (but blocked) in AS': if $\neg q_2 \in \mathcal{K}'$ then the argument for t is attacked by the argument based on $\neg q_2 \Rightarrow l_1$; if $\neg q_2 \notin \mathcal{K}'$ then the argument for t is attacked by the argument based on $q_2 \Rightarrow \neg t$.

3. *If there is an argument A for l in $Arg(AS)$ and there is no observation-based potential argument in $P(AS)$ that p-attacks A, then $\neg u(l)$ and $\neg o(l)$.*

4. *If each potential argument A^p for l in $P(AS)$ is p-attacked by an argument B in $Arg(AS)$ and there is no observation-based potential argument in $P(AS)$ that p-attacks B, then $\neg d(l)$.*

5. *If there is an argument A for l in $Arg(AS)$ and each potential argument B^p in $P(AS)$ that p-attacks A is p-attacked by an observation-based argument in $Arg(AS)$, then $L'(l) = \langle 0,1,0,0 \rangle$.*

Proof. Next we prove the five parts of the lemma individually:

1. **If there is an argument for l in $Arg(AS)$, then $\neg u(l)$.**

   Suppose that there is an argument for l in $Arg(AS)$. Then by Lemma 1, there is a non-circular argument for l in $Arg(AS)$. Let A be a non-circular argument for l in $Arg(AS)$. A is non-circular, so by Lemma 2, $\mathtt{h}(A)$ is finite. Next, we prove that l is labelled $\neg u(l)$ by induction on the height of A.

   Base case: If $\mathtt{h}(A) = 0$, then $l \in \mathcal{K}$ and l is labelled $\neg u(l)$ by Definition 9 case L-U-a.

   Induction hypothesis: If $\mathtt{h}(A) \leq k$, then l is labelled $\neg u(l)$ by L'.

   Induction step: If $\mathtt{h}(A) = k+1$, then there is a rule r for l such that for each $a \in \mathtt{ants}(r)$, there is an argument A_i for a in $Arg(AS)$ with $\mathtt{h}(A_i) \leq k$. By the induction hypothesis, there is a rule r for l such that for each $a \in \mathtt{ants}(r)$, a is labelled $\neg u(a)$ by L'. Then by case R-U-a, r is labelled $\neg u(r)$ and by case L-U-b, l is labelled $\neg u(l)$ by L'.

2. **If each potential argument for l in $P(AS)$ is p-attacked by an observation-based argument in $Arg(AS)$, then $\neg d(l)$ and $\neg b(l)$.**

   Suppose that each potential argument for l in $P(AS)$ is p-attacked by an observation-based argument in $Arg(AS)$. If there is no potential argument for l in $P(AS)$, then by Lemma 15, $L_p(l) = \langle 1,0,0,0 \rangle$, which implies that $L'(l) = \langle 1,0,0,0 \rangle$, so l is labelled $\neg d(l)$ and $\neg b(l)$.

   Alternatively, there is a potential argument for l in $P(AS)$. Then by Lemma 13, there is a non-circular potential argument for l in $P(AS)$. Let A^p be an arbitrary non-circular potential argument for l in $P(AS)$. A^p is non-circular, so by Lemma 14 $\mathtt{h}(A^p)$ is finite. By assumption, A^p is p-attacked by an observation-based argument in $Arg(AS)$. We prove that l is labelled $\neg d(l)$ and $\neg b(l)$ by induction on the height of l. We take $\mathtt{h}(A^p) = 1$ as base case: if $\mathtt{h}(A^p) = 0$, then A^p cannot be p-attacked on any other subargument than its conclusion. So A^p must be attacked by an argument for $-l$. We assumed that A^p is attacked by an observation-based argument, so $-l \in \mathcal{K}$. However, this contradicts

the assumption that A^p is a potential argument with $\mathtt{h}(A^p) = 0$: by Definition 10, this would require that $-l \notin \mathcal{K}$.

Base case: Suppose that $\mathtt{h}(A^p) = 1$.

First suppose that $l \notin \mathcal{Q}$. This means that A^p cannot be p-attacked by an observation-based argument in $Arg(AS)$ *on its conclusion*. Therefore A^p must be p-attacked by an observation-based argument in $Arg(AS)$ *on a subargument*. A^p has the form $A_1,\ldots,A_m \Rightarrow c$ and for each $i \in [1\,..\,m]$: $\mathtt{h}(A_i) = 0$, so for each $a \in \mathtt{ants}(r)$: $a \in \mathcal{Q}$ and $-a \notin \mathcal{K}$. Given that for each $a \in \mathtt{ants}(r)$: $-a \notin \mathcal{K}$, A^p cannot be p-attacked by observation-based argument in $Arg(AS)$; this contradicts the assumption that A^p is p-attacked by an observation-based argument in $Arg(AS)$.

As a result, l must be in $\mathcal{Q}$. We prove by contradiction that $-l \in \mathcal{K}$. If $l \in \mathcal{Q}$ and $-l \notin \mathcal{K}$ then there is an observation-based potential argument for l in $P(AS)$, which is not p-attacked by any observation-based argument in $Arg(AS)$ since $-l \notin \mathcal{K}$, a contradiction. Given that $-l \in \mathcal{K}$, l is labelled $\neg d(l)$ by L-D-a and $\neg b(l)$ by L-B-a.

Induction hypothesis: If $\mathtt{h}(A^p) \leq k$, then l is labelled $\neg d(l)$ and $\neg b(l)$ by L'.

Induction step: Suppose that $\mathtt{h}(A^p) = k+1$. If $l \in \mathcal{Q}$, then $-l \in \mathcal{K}$. We prove this by contradiction: if $l \in \mathcal{Q}$ and $-l \notin \mathcal{K}$ then there is an observation-based potential argument for l in $P(AS)$, which is not p-attacked by any observation-based argument in $Arg(AS)$ since $-l \notin \mathcal{K}$. Given that $-l \in \mathcal{K}$, l is labelled $\neg d(l)$ by L-D-a and $\neg b(l)$ by L-B-a.

Alternatively, suppose that $l \notin \mathcal{Q}$. Then each potential argument for l in $P(AS)$ must be rule-based. Let r be an arbitrary rule for l. Then each potential argument based on r in $P(AS)$ is p-attacked by an observation-based argument in $Arg(AS)$. Let A^p be an arbitrary argument based on r in $P(AS)$. A^p must be p-attacked by an observation-based argument in $Arg(AS)$ on a subargument, since $l \notin \mathcal{Q}$. So there is some $a \in \mathtt{ants}(r)$ such that each potential argument for a in $P(AS)$ is p-attacked by an observation-based argument in $Arg(AS)$. Then by the induction hypothesis, this antecedent a is labelled $\neg d(a)$ and $\neg b(a)$. Then by Definition 9 case R-D-a and R-B-b, r is labelled $\neg d(r)$ and $\neg b(r)$ by L'. We chose r as an arbitrary rule for l, so each rule r for l is labelled $\neg d(r)$ and $\neg b(r)$ by L'. Then by Definition 9 case L-D-b and L-B-b, l is labelled $\neg d(l)$ and $\neg b(l)$ by L'.

3. **If there is an argument A for l in $Arg(AS)$ and there is no observation-based potential argument in $P(AS)$ that p-attacks A, then $\neg u(l)$ and $\neg o(l)$.**

   Suppose that there is an argument A^{circ} for l in $Arg(AS)$ that is not p-attacked by any observation-based potential argument in $P(AS)$.

   Following the proof of Lemma 1, a non-circular argument A for l can be constructed from A^{circ} such that if A is attacked then A^{circ} is attacked. Hence A is not p-attacked by any observation-based potential argument in P(AS) either.

   A is non-circular, so by Lemma 2, $\mathtt{h}(A)$ is finite. Next, we prove that l is labelled $\neg u(l)$ and $\neg o(l)$ by induction on the height of A.

   Base case: If $\mathtt{h}(A) = 0$, then $l \in \mathcal{K}$ and l is labelled $\neg u(l)$ by Definition 9 case L-U-a and $\neg o(l)$ by case L-O-a.

   Induction hypothesis: If $\mathtt{h}(A) = k$, then l is labelled $\neg u(l)$ and $\neg o(l)$ by L'.

   Induction step: If $\mathtt{h}(A) = k+1$, then there is a rule r for l such that for each antecedent $a \in \mathtt{ants}(r)$, there is an argument A_i for a in $Arg(AS)$, such that A_i is not p-attacked by any observation-based potential argument in $P(AS)$ and $\mathtt{h}(A_i) \leq k$. Then by the induction hypothesis, each $a \in \mathtt{ants}(r)$ is labelled $\neg u(a)$ and $\neg o(a)$ by L'. Then by Definition 9 case R-U-a and R-O-a, r is labelled $\neg u(r)$ and $\neg o(r)$ by L'. Given that there is a rule r for l that is labelled $\neg u(r)$, l is labelled $\neg u(l)$ by case L-U-b. If $l \notin \mathcal{Q}$, then l is labelled $\neg o(l)$ by case L-O-d. Alternatively, if $l \in \mathcal{Q}$, then $l \in \mathcal{K}$. We prove this by contradiction: if $l \notin \mathcal{K}$, then there is an observation-based potential argument for $-l$ in $P(AS)$ which p-attacks A and that would contradict our assumption on l. Since $l \in \mathcal{K}$, l is labelled $\neg o(l)$ by case L-O-a.

4. **If each potential argument A^p for l in $P(AS)$ is p-attacked by an argument B in $Arg(AS)$ and there is no observation-based potential argument in $P(AS)$ that p-attacks B, then $\neg d(l)$.**

   Suppose that each potential argument A^p for l in $P(AS)$ is p-attacked by an argument in $Arg(AS)$ that is not p-attacked by any observation-based potential argument in $P(AS)$.

   If there is no potential argument for l in $P(AS)$, then by Lemma 15, $L_p(l) = \langle 1,0,0,0\rangle$. This implies that $L'(l) = \langle 1,0,0,0\rangle$, so l is labelled $\neg d(l)$ by L'.

   Alternatively, there is a potential argument for l in $P(AS)$. We distinguish two situations: either $l \in \mathcal{Q}$ or $l \notin \mathcal{Q}$.

   - If $l \in \mathcal{Q}$ then $-l \in \mathcal{K}$: we prove this by contradiction. Suppose that $l \in \mathcal{Q}$ and $-l \notin \mathcal{K}$. Then there is an observation-based potential argument for l. This argument can only be p-attacked by an observation-based potential argument for $-l$, but given that $-l \notin \mathcal{K}$, this observation-based potential argument is p-attacked by the observation-based potential argument for l, which contradicts our assumption. So $-l \in \mathcal{K}$. Then l is labelled $\neg d(l)$ by Definition 9 case L-D-a.
   - Alternatively, $l \notin \mathcal{Q}$.
     - First suppose that there is an argument B for $-l$ in $Arg(AS)$ that is not p-attacked by any observation-based potential argument in $P(AS)$. B cannot be observation-based, since $-l \notin \mathcal{Q}$, so there is a rule r' for $-l$ such that for each $a' \in \mathtt{ants}(r')$: there is an argument for a' in $Arg(AS)$ that is not p-attacked by any observation-based potential argument in $P(AS)$. Then by item 3 of this lemma, for each $a' \in \mathtt{ants}(r')$, a' is labelled $\neg u(a')$ and $\neg o(a')$. This implies that r' is labelled $\neg u(r')$ by Definition 9 case R-U-a and $\neg o(r')$ by case R-O-a. Given that $-l \notin \mathcal{Q}$ and there is a rule r' for $-l$ that is labelled $\neg u(r')$ and $\neg o(r')$, we derive that l is labelled $\neg d(l)$ by case L-D-c.
     - Alternatively, there is no argument for $-l$ in $Arg(AS)$ that is not p-attacked by any observation-based potential argument in $P(AS)$. Still there must be some argument in $Arg(AS)$ that attacks A^p and is not attacked by any observation-based potential argument in $P(AS)$. Given that $l \notin \mathcal{Q}$, we derive that each potential argument for l in $P(AS)$ must be rule-based and p-attacked *on a subargument* by an argument in $Arg(AS)$ that is not p-attacked by any observation-based potential argument in $P(AS)$. To show this, we will first show that each rule r for l in $\mathcal{R}$ is labelled $\neg d(r)$.
       Let r be an arbitrary rule for l in $\mathcal{R}$. We consider two cases:
       * First suppose that there is no potential argument based on r in $P(AS)$. Then by Lemma 15, r is labelled $\langle 1,0,0,0\rangle$ in the preprocessing step. Hence r is labelled $\neg d(r)$.
       * Alternatively, suppose that there is a potential argument based on r in $P(AS)$. By Definition 10 of potential arguments, this means that for each $a \in \mathtt{ants}(r)$, there is a potential argument for a in $P(AS)$. Then by Lemma 13, for each $a \in \mathtt{ants}(r)$, there is a non-circular potential argument for a in $P(AS)$; as a result, there is a non-circular potential argument based on r in $P(AS)$. Let A^p be an arbitrary non-circular potential argument based on r in $P(AS)$. A^p is non-circular, so by Lemma 14, $\mathtt{h}(A^p)$ is finite. We prove by induction on the height of A^p that r (its top rule) is labelled $\neg d(r)$. Note that the height of A^p must be at least 1 since we assumed that A^p is based on r.
         Base case: If $\mathtt{h}(A^p) = 1$, then A^p has the form $A_1,\ldots,A_m \Rightarrow l$ and for each $i \in [1\,..\,m]$: $\mathtt{h}(A_i) = 0$. By Definition 10 of potential arguments, this implies that for each $a \in \mathtt{ants}(r)$: $a \in \mathcal{Q}$.
         Remember A^p is p-attacked by an argument in $Arg(AS)$ that is not p-attacked by any observation-based potential argument in $P(AS)$, and that this argument is not an argument for $-l$. This means that A^p must be p-attacked *on a subargument* by an argument in $Arg(AS)$ that is not p-attacked by any observation-based potential argument in $P(AS)$. As a result, there must be some $a \in \mathtt{ants}(r)$ such that there is an argument for $-a$ in $Arg(AS)$ that is not p-attacked by any observation-based potential argument in $P(AS)$. Since $\mathtt{h}(A^p) = 1$, we know that for each $i \in [1\,..\,m]$: $\mathtt{h}(A_i) = 0$, so for each

$a \in \texttt{ants}(r)$: $a \in \mathcal{Q}$. Now let a be the antecedent of $\texttt{ants}(r)$ such that there is an argument for $-a$ in $Arg(AS)$ that is not p-attacked by any observation-based potential argument in $P(AS)$. If $-a \notin \mathcal{K}$, then each argument for $-a$ would be p-attacked by an observation-based potential argument for a in $P(AS)$ which would contradict our earlier assumption, so $-a \in \mathcal{K}$. Then a is labelled $\neg d(a)$ by Definition 9 case L-D-a. Given that a is an antecedent of r, r must have been labelled $\neg d(r)$ by Definition 9 case R-D-a.

Induction hypothesis: If $\texttt{h}(A^p) \leq k$ then r is labelled $\neg d(r)$.

Induction step: Suppose that $\texttt{h}(A^p) = k+1$. Then A^p has the form $A_1, \ldots, A_m \Rightarrow l$ and for each $i \in [1 \,..\, m]$: $A_i \in P(AS)$ and $\texttt{h}(A_1) \leq k$. Furthermore, there is some $i \in [1 \,..\, m]$ such that A_i is p-attacked by an argument in $Arg(AS)$ that is not p-attacked by any observation-based potential argument in $P(AS)$. Then there is at least one $a \in \texttt{ants}(r)$ such that each potential argument for a is p-attacked by an argument B in $Arg(AS)$ and there is no observation-based potential argument in $P(AS)$ that p-attacks B. So by the induction hypothesis, this a is labelled $\neg d(a)$. Then by Definition 9 case R-D-a, r is labelled $\neg d(r)$.

We just showed that r is labelled $\neg d(r)$ in both cases.

Finally, since we picked r as an arbitrary rule for l, this implies that each rule r for l is labelled $\neg d(r)$ so by Definition 9 case L-D-b, l is labelled $\neg d(l)$.

To conclude, we have shown that for each l (both if $l \in \mathcal{Q}$ and if $l \notin \mathcal{Q}$): if each potential argument A^p for l in $P(AS)$ is p-attacked by an argument B in $Arg(()AS)$ and there is no observation-based potential argument in $P(AS)$ that p-attacks B, then l is labelled $\neg d(l)$.

5. **If there is an argument A for l in $Arg(AS)$ and each potential argument B^p in $P(AS)$ that p-attacks A is p-attacked by an observation-based argument in $Arg(AS)$, then $L'(l) = \langle 0,1,0,0 \rangle$.**

   Suppose that there is an argument A^{circ} for l in $Arg(AS)$ such that each potential argument p-attacking A^{circ} is p-attacked by an observation-based argument in $Arg(AS)$.

   Following the proof of Lemma 1, a non-circular argument A for l can be constructed from A^{circ} such that if A is attacked then A^{circ} is attacked. Hence any potential argument that p-attacks A p-attacks A^{circ} as well and is therefore p-attacked by an observation-based argument in $Arg(AS)$.

   A is non-circular, so by Lemma 2, $\texttt{h}(A)$ is finite. Next, we prove that $L'(l) = \langle 0,1,0,0 \rangle$ by induction on the height of A.

   Base case: If $\texttt{h}(A) = 0$, then $l \in \mathcal{K}$. Then l is labelled $L'(l) = \langle 0,1,0,0 \rangle$ by case L-U-a, L-O-a and L-B-a.

   Induction hypothesis: If $\texttt{h}(A) \leq k$, then $L'(l) = \langle 0,1,0,0 \rangle$.

   Induction step: Suppose that $\texttt{h}(A) = k+1$. We consider two cases: either $l \in \mathcal{Q}$ or $l \notin \mathcal{Q}$.

   - Suppose that $l \in \mathcal{Q}$. We now show that $l \in \mathcal{K}$. Suppose, towards a contradiction, that $l \notin \mathcal{K}$. Then there is an observation-based potential argument for $-l$ in $P(AS)$. The potential argument $-l$ p-attacks each argument for l in $Arg(AS)$ (including A), but is not attacked by any observation-based argument in $Arg(AS)$ since $l \notin \mathcal{K}$. This contradicts our initial assumption, so if $l \in \mathcal{Q}$ then $l \in \mathcal{K}$. Then l is labelled $L'(l) = \langle 0,1,0,0 \rangle$ by case L-U-a, L-O-a and L-B-a.
   - Alternatively, suppose that $l \notin \mathcal{Q}$. First we show that each rule r' for $-l$ (if any) is labelled $\neg d(r')$ and $\neg b(r')$. Let $r' \in \mathcal{R}$ be an arbitrary rule for $-l$.
     - If there is no potential argument based on r' in $P(AS)$, then by Lemma 15 $L'(r') = \langle 1,0,0,0 \rangle$. This means that r' is labelled $\neg d(r')$ and $\neg b(r')$.
     - Otherwise, there is a potential argument based on r' in $P(AS)$. Then by Definition 10: for each $a' \in \texttt{ants}(r')$: there is a potential argument for a' in $P(AS)$.
       Next, we prove by contradiction that there must be some antecedent $a' \in \texttt{ants}(r')$ such that each potential argument for a' is p-attacked by an observation-based argument in

$Arg(AS)$. Suppose that for each $a' \in \texttt{ants}(r')$: there is a potential argument for a' in $P(AS)$ that is not p-attacked by any observation-based argument. Then we could construct from these potential arguments a potential argument B^p based on r' (hence p-attacking A: $\texttt{conc}(r') = -\texttt{conc}(A)$) that is not p-attacked by any observation-based argument in $Arg(AS)$. Remember our assumption that $l \notin \mathcal{Q}$; this implies that $-l \notin \mathcal{Q}$, so B^p cannot be p-attacked on its conclusion by an observation-based argument in $Arg(AS)$. However, this contradicts our assumption that each potential argument in $P(AS)$ that p-attacks A is p-attacked by an observation-based argument in $Arg(AS)$. To conclude, there must be some antecedent $a' \in \texttt{ants}(r')$ such that each potential argument for a' is p-attacked by an observation-based argument in $Arg(AS)$.

Then by item 2 of this lemma, there must be some antecedent $a' \in \texttt{ants}(r')$ that is labelled $\neg d(a')$ and $\neg b(a')$ by L'. This implies that r' is labelled $\neg d(r')$ and $\neg b(r')$ by L' (Definition 9 case R-D-a and R-B-b).

We chose r' as an arbitrary rule for $-l$, so we can conclude that each rule r' for $-l$ in $\mathcal{R}$ is labelled $\neg d(r')$ and $\neg b(r')$.

Next, we will prove that for at least one rule r for l in $\mathcal{R}$, each antecedent $a \in \texttt{ants}(r)$ should be labelled $L'(a) = \langle 0,1,0,0\rangle$. We will prove this by contradiction. Suppose that for each rule r for l there is an antecedent $a \in \texttt{ants}(r)$ such that $L'(a) \neq \langle 0,1,0,0\rangle$. Let r be an arbitrary rule for l and let a be an arbitrary antecedent in $\texttt{ants}(r)$ such that $L'(a) \neq \langle 0,1,0,0\rangle$. Then by the induction hypothesis, there is no argument A' for a such that each potential argument in $P(AS)$ that p-attacks A' is p-attacked by an observation-based argument in $P(AS)$. This implies that there is no argument A based on r such that each potential argument in $P(AS)$ that p-attacks A is p-attacked by an observation-based argument in $P(AS)$. Since we chose r as an arbitrary rule for l, there is no rule-based argument A for l in $Arg(AS)$ such that each potential argument in $P(AS)$ that p-attacks A is p-attacked by an observation-based argument in $P(AS)$. Note that there is no observation-based argument for l either ($l \notin \mathcal{Q}$), so there is no argument A for l such that each potential argument in $P(AS)$ that p-attacks A' is p-attacked by an observation-based argument in $P(AS)$. This contradicts our initial assumption, so for at least one rule r for l in $\mathcal{R}$, each antecedent $a \in \texttt{ants}(r)$ should be labelled $L'(a) = \langle 0,1,0,0\rangle$. Then by Definition 9 case R-U-a, R-O-b and R-B-b, there is a rule r for l in $\mathcal{R}$ that is labelled $L'(r) = \langle 0,1,0,0\rangle$.

In summary, we have that if $l \notin \mathcal{Q}$: each rule r' for $-l$ in $\mathcal{R}$ is labelled $\neg d(r')$ and $\neg b(r')$ and there is a rule r for l in $\mathcal{R}$ that is labelled $L'(r) = \langle 0,1,0,0\rangle$. Then by Definition 9 case L-U-b, L-O-b and L-B-d, l is labelled $L'(l) = \langle 0,1,0,0\rangle$.

To conclude: if there is an argument A for l and each potential argument in $P(AS)$ that p-attacks A is p-attacked by an observation-based argument in $Arg(AS)$, then l is labelled $L'(l) = \langle 0,1,0,0\rangle$. □

In Example 12 we saw an argumentation setup $AS = (\mathcal{L}, \mathcal{R}, \mathcal{Q}, \mathcal{K})$ in which STABILITY did not detect that a literal $l \in \mathcal{L}$ was not out or unsatisfiable in each future $AS' \in F(AS)$. The next lemma shows that this issue only occurs if l is inconsistently supported in AS.

Lemma 17 (Incorrect labelling $u(l)$ or $o(l)$ due to inconsistent support)**.** *Given an argumentation setup $AS = (\mathcal{L}, \mathcal{R}, \mathcal{Q}, \mathcal{K})$ and labelling L' after executing the* STABILITY *algorithm. Given a literal $l \in \mathcal{L}$, if for each $AS' \in F(AS)$, l is not unsatisfiable or out in AS', but l is labelled $u(l) \vee o(l)$ by L', then l is inconsistently supported in AS.*

Proof. Let $l \in \mathcal{L}$ be a literal such that for each $AS' \in F(AS)$, l is not unsatisfiable or out in AS'. Suppose that l is labelled $u(l)$ or $o(l)$ by L'. Then l is not labelled ($\neg u(l)$ and $\neg o(l)$), so by Lemma 16.3 there is no argument for l in $Arg(AS)$ that is not p-attacked by an observation-based potential argument in $P(AS)$. In other words, each argument for l in $Arg(AS)$ is p-attacked by an observation-based potential argument in $P(AS)$.

We assumed that l is not unsatisfiable or out in any $AS' \in F(AS)$, so l is not unsatisfiable or out in AS itself. That means that there must be at least one argument for l in $Arg(AS)$ that is not attacked by an argument in the grounded extension. Let S be the set of arguments for l in $Arg(AS)$ that are not attacked by

an argument in the grounded extension $G(AS)$ - intuitively, each argument $A \in S$ justifies the assumption that l is not unsatisfiable or out in AS. By Lemma 4, for each $A \in S$: A is not attacked by an argument in $G(AS)$ iff for each $A' \in \texttt{sub}(A)$: $-\texttt{conc}(A') \notin \mathcal{K}$. Then:

$$S = \{A \mid A \in Arg(AS),\ \texttt{conc}(A) = l,\ \text{for each } A' \in \texttt{sub}(A) : -\texttt{sub}(A') \notin \mathcal{K}\}$$

In combination with our earlier conclusion that each argument for l in $Arg(AS)$ is p-attacked by an observation-based potential argument in $P(AS)$, we now have that for each $A \in S$, there is a subargument $A' \in \texttt{sub}(A)$ such that $\texttt{conc}(A') \in \mathcal{Q}$, $\texttt{conc}(A') \notin \mathcal{K}$ and $-\texttt{conc}(A') \notin \mathcal{K}$.

Now construct the set consisting of all observation-based potential arguments attacking an argument A for l in S:

$$\mathcal{K}^{Att} = \{-a \mid\ A \in S,\ A' \in \texttt{sub}(A), \texttt{conc}(A') = a, a \in \mathcal{Q},\ a \notin \mathcal{K},\ -a \notin \mathcal{K}\}$$

Suppose that l is **not** inconsistently supported in AS. Then by Definition 12 of inconsistent support, there is no $A^p, B^p \in P(AS)$ such that $\texttt{conc}(A^p) = \texttt{conc}(B^p) = l$ and A^p is inconsistent with B^p. As a result, the set of all premises of all potential arguments for l K^{Sup} must be consistent, where:

$$\mathcal{K}^{Sup} = \{p \mid A^p \in P(AS), \texttt{conc}(A^p) = l, p \in \texttt{prem}(A^p)\}$$

We will use this consistency property to construct a future argumentation setup AS^{Att}.

Given that $\mathcal{K}^{Sup}$ is consistent, the set $\mathcal{K}^{NegSup} = \{-p \mid A^p \in P(AS), \texttt{conc}(A^p) = l, p \in \texttt{prem}(A^p)\}$ must be consistent as well.

We now show that $\mathcal{K}^{Att}$ is a subset of $\mathcal{K}^{NegSup}$: for each $A \in S$: A is an argument for l, so A also is a potential argument for l. This implies that each element in $\mathcal{K}^{Att}$ is the negation of a premise of a potential argument for l; $\mathcal{K}^{NegSup}$ contains the negation of each premise of a potential argument for l. Therefore $\mathcal{K}^{Att} \subseteq \mathcal{K}^{NegSup}$. Given that $\mathcal{K}^{Att} \subseteq \mathcal{K}^{NegSup}$ and $\mathcal{K}^{NegSup}$ is consistent, $\mathcal{K}^{Att}$ must be consistent as well.

Furthermore, $\mathcal{K}^{Att}$ is defined in such a way that it is consistent with $\mathcal{K}$ (because $a \notin \mathcal{K}$ and Definition 1 states that $\mathcal{K}$ is consistent itself). As a result, $\mathcal{K} \cup \mathcal{K}^{Att}$ must be consistent. Furthermore, $\mathcal{K} \subseteq \mathcal{K}^{Att} \subseteq \mathcal{Q}$, so the argumentation setup $AS^{Att} = (\mathcal{L}, \mathcal{R}, \mathcal{Q}, \mathcal{K} \cup \mathcal{K}^{Att})$ is in $F(AS)$. Note that, because of the way we defined AS^{Att}, each potential argument in $P(AS)$ attacking an $A \in S$ is an argument in $Arg(AS^{Att})$. As a result, each argument in $Arg(AS^{Att})$ that was already in S is attacked by an observation-based argument in $Arg(AS^{Att})$.

Remember that we assumed that for each $AS' \in F(AS)$, l is not unsatisfiable or out in AS'. $AS^{Att} \in F(AS)$, so l is not unsatisfiable or out in AS^{Att}. By Definition 6, there is an argument for l in $Arg(AS^{Att})$ that is not attacked by an argument in the grounded extension $G(AS^{Att})$. By Lemma 4, there is an argument B for l in $Arg(AS^{Att})$ that is not attacked by an observation-based argument in $Arg(AS^{Att})$. However, $B \notin Arg(AS)$, as we show next. B is not attacked by an observation-based argument in $Arg(AS^{Att})$, so B was not attacked by an observation-based potential argument in $Arg(AS)$. If B would have been in $Arg(AS)$, then B would have been part of S; we just derived that each argument in $Arg(AS^{Att})$ that was already in S is attacked by an observation-based argument in $Arg(AS^{Att})$ (which is not the case for B).

Given that $B \notin Arg(AS)$, there must be some $b \in \texttt{prem}(B)$ that is not in $\mathcal{K}$ (but in $\mathcal{K}^{Att}$). So there is some $b \in \texttt{prem}(B)$ such that there is some argument A for l in S that has a subargument A' with $\texttt{conc}(A') = b$, $b \in \mathcal{Q}$, $b \notin \mathcal{K}$ and $-b \notin \mathcal{K}$. But then there also must be a potential argument $A^p \in P(AS)$ which equals A, except for the fact that the subargument A' is replaced by the observation-based potential argument $-b$. However, this means that B is a potential argument for l in $P(AS)$ with $b \in \texttt{prem}(B)$ and A^p is a potential argument for l in $P(AS)$ with $-b \in \texttt{prem}(A^p)$. In other words: l is inconsistently supported in AS. □

In Example 13 we saw an argumentation setup $AS = (\mathcal{L}, \mathcal{R}, \mathcal{Q}, \mathcal{K})$ in which STABILITY did not detect that a literal $l \in \mathcal{L}$ was not defended or blocked in each future $AS' \in F(AS)$. The next lemma shows that this issue only occurs if l is inconsistently supported in AS.

Lemma 18 (Incorrect labelling $d(l)$ or $b(l)$ due to inconsistent support)**.** *Given an argumentation setup $AS = (\mathcal{L}, \mathcal{R}, \mathcal{Q}, \mathcal{K})$ and labelling L' after executing the* STABILITY *algorithm. Given a literal $l \in \mathcal{L}$, if for each $AS' \in F(AS)$, l is not defended or blocked in AS', but l is labelled $d(l) \vee b(l)$ by L', then l is inconsistently supported in AS.*

Proof. Let $l \in \mathcal{L}$ be a literal and suppose that for each $AS' \in F(AS)$, l is not defended or blocked in AS' and l is labelled $d(l) \vee b(l)$ by L'. By Lemma 16.2, if l is labelled $d(l) \vee b(l)$ then there is a potential argument A^p for l in $P(AS)$ that is not p-attacked by an observation-based argument B in $Arg(AS)$.

Let $A^p \in P(AS)$ be an arbitrary potential argument for l such that A^p is not attacked by an observation-based B in $Arg(AS)$. Suppose that l is not inconsistently supported in AS; then $\texttt{prem}(A^p)$ must be consistent: if $a \in \texttt{prem}(A^p)$ then $-a \notin \texttt{prem}(A^p)$. Since $A^p \in P(AS)$, we have that if $a \in \texttt{prem}(A^p)$ then $-a \notin \mathcal{K}$. Let $\mathcal{K}^{Sup} = \mathcal{K} \cup \texttt{prem}(A^p)$. For each $a \in \mathcal{K}^{Sup} : -a \notin \mathcal{K}^{Sup}$, so $AS^{Sup} = (\mathcal{L}, \mathcal{R}, \mathcal{Q}, \mathcal{K}^{Sup})$ is an argumentation setup. Since $\mathcal{K} \subseteq \mathcal{K}^{Sup} \subseteq \mathcal{Q}$, $AS^{Sup} \in F(AS)$.

We assumed that for each $AS' \in F(AS)$, l is not defended or blocked in AS' and $AS^{Sup} \in F(AS)$, so l is not defended or blocked in AS^{Sup}. However, for each $a \in \texttt{prem}(A^p)$, $a \in \mathcal{K}^{Sup}$ so A^p is an argument for l in $Arg(AS^{Sup})$. Then by Definition 6, l is not unsatisfiable either in AS^{Sup}. So l must be out in AS^{Sup}; by Lemma 4 each argument for l must be attacked by an observation-based argument in $Arg(AS^{Sup})$. We know that A^p is not attacked by an observation-based argument B in $Arg(AS)$. Formally: for each $A' \in \texttt{sub}(A^p) : -\texttt{conc}(A') \notin \mathcal{K}$. So there must be some $A'' \in \texttt{sub}(A^p)$ such that $-\texttt{conc}(A'') \in \mathcal{K}^{Sup} - \mathcal{K} = \texttt{prem}(A^p)$. Let $a = \texttt{conc}(A'')$. But then $-a \in \mathcal{Q}$ and $-a \notin \mathcal{K}$, so there must be some potential argument $A^\perp \in P(AS)$, where $A^\perp$ is the same as A^p but A'' is replaced by the observation a. However, this implies that $a \in \texttt{prem}(A^\perp)$ and $-a \in \texttt{prem}(A^\perp)$, so $A^\perp$ is inconsistent with $A^\perp$. To conclude, l is inconsistently supported in AS (so l is inconsistently supported or attacked in AS). □

Finally, we use Lemma 16–18 to show that, given an argumentation setup $AS = (\mathcal{L}, \mathcal{R}, \mathcal{Q}, \mathcal{K})$ and a literal $l \in \mathcal{L}$ such that l is stable in AS, L' labels l as being stable, unless l is inconsistently supported or attacked in AS.

Proposition 3 (Conditional completeness stability labelling)**.** *Given an argumentation setup $AS = (\mathcal{L}, \mathcal{R}, \mathcal{Q}, \mathcal{K})$ and labelling L' after executing the* STABILITY *algorithm. Given a literal $l \in \mathcal{L}$, if l is stable in AS but l is not labelled stable by L', then l is inconsistently supported or attacked in AS.*

Proof. Given an argumentation setup $AS = (\mathcal{L}, \mathcal{R}, \mathcal{Q}, \mathcal{K})$ and labelling L' after executing the STABILITY algorithm. Let $l \in \mathcal{L}$ be a literal.

Unsatisfiable Suppose that for each $AS' \in F(AS)$, l is unsatisfiable in AS' and $L'(l) \neq \langle 1,0,0,0 \rangle$. Then the preprocessessing step must have been labelled $L_p(l) \neq \langle 1,0,0,0 \rangle$. So by Lemma 15, there is a potential argument $A^p \in P(AS)$ with $\texttt{conc}(A^p) = l$. We will now prove by contradiction that $\texttt{prem}(A^p)$ is inconsistent. Suppose that $\texttt{prem}(A^p)$ is consistent (that is: for each $a \in \texttt{prem}(A^p) : -a \notin \texttt{prem}(A^p)$). This would imply that the set $\mathcal{K}^A = \mathcal{K} \cup \texttt{prem}(A^p)$ is consistent, since $\texttt{prem}(A^p)$ is required to be consistent with $\mathcal{K}$ by Definition 10. Then $AS^A = (\mathcal{L}, \mathcal{R}, \mathcal{Q}, \mathcal{K}^A)$ is an argumentation setup. $\mathcal{K} \subseteq \mathcal{K}^A \subseteq \mathcal{Q}$, so $AS^A \in F(AS)$. We supposed that for each $AS' \in F(AS)$, l is unsatisfiable in AS', but this contradicts with the fact that $A^p \in Arg(AS^A)$. Therefore we have to retract our assumption that $\texttt{prem}(A^p)$ is consistent. To conclude, A^p is inconsistent with itself, hence l is inconsistently supported in AS (so l is inconsistently supported or attacked in AS).

Out Suppose that for each $AS' \in F(AS)$, l is out in AS' and $L'(l) \neq \langle 0,0,1,0 \rangle$. $AS \in F(AS)$, so l is out in $F(AS)$ and l is not unsatisfiable in $F(AS)$. Then by Lemma 16.1, l is labelled $\neg u(l)$ by L'. So l must be labelled $d(l) \vee b(l)$ by L', although for each $AS' \in F(AS)$, l is not defended or blocked in AS'. Then by Lemma 18, l is inconsistently supported or attacked in AS.

Defended Suppose that for each $AS' \in F(AS)$, l is defended in AS' and $L'(l) \neq \langle 0,1,0,0 \rangle$. We consider both the case that l is labelled $u(l) \vee o(l)$ and the alternative case that l is labelled $\neg u(l) \wedge \neg o(l)$ by L'.

- First suppose that l is labelled $u(l) \vee o(l)$ by L'. Since l is defended in AS' for each $AS' \in F(AS)$, we know that for each $AS' \in F(AS)$, l is not unsatisfiable or out in AS'. Then by Lemma 17, l is inconsistently supported in AS.
- Alternatively, l is labelled $\neg u(l) \wedge \neg o(l)$ by L' but still $L' \neq \langle 0,1,0,0 \rangle$.

By Lemma 16.5, the fact that $L'(l) \neq \langle 0,1,0,0\rangle$ implies that for each argument A for l in $Arg(AS)$, there is some potential argument in $P(AS)$ that p-attacks A but is not p-attacked by any observation-based argument in $Arg(AS)$.

We assumed that l is labelled $\neg u(l)$, so by Definition 9 case L-U-a/b, either $l \in \mathcal{K}$ or there is a rule r for l that is labelled $\neg u(r)$. However, if $l \in \mathcal{K}$, then $l \in \mathcal{Q}$, so l would be labelled $\neg b(l)$ by Definition 9 case L-B-a. That would mean that $L'(l) = \langle 0,1,0,0\rangle$, which contradicts our assumption. So we need to retract our assumption that $l \in \mathcal{K}$ and derive that there is a rule r for l that is labelled $\neg u(r)$. Then by Lemma 9, there is a rule-based argument for l in $Arg(AS)$. Let A be an arbitrary argument for l in $Arg(AS)$. Then there is some potential argument in $P(AS)$ that p-attacks A but is not p-attacked by any observation-based argument in $Arg(AS)$. Let B^p be an arbitrary potential argument in $P(AS)$ that p-attacks A but is not p-attacked by any observation-based argument in $Arg(AS)$.

We now prove by contradiction that l is inconsistently attacked in AS. Suppose that l is not inconsistently attacked in AS. Then B^p must be consistent with itself, so there is an argumentation setup $AS^B = (\mathcal{L}, \mathcal{R}, \mathcal{Q}, \mathcal{K}^B)$ where $\mathcal{K}^B = \mathcal{K} \cup \texttt{prem}(B^p)$. Note that $B^p \in Arg(AS^B)$ and $AS^B \in F(AS)$. B^p is not attacked by an observation-based argument in $Arg(AS)$, so for each $B'' \in \texttt{sub}(B^p)$: $-\texttt{conc}(B'') \notin \mathcal{K}$. Given that l is defended in AS^B, there must be some $B' \in \texttt{sub}(B^p)$ such that $-\texttt{conc}(B') \in \texttt{prem}(B^p)$. Let $b = \texttt{conc}(B')$. But that would imply that there is some potential argument $B^\perp \in P(AS)$, where $B^\perp$ is the same as B^p but B' is replaced by the observation b. However, this implies that $b \in \texttt{prem}(B^\perp)$ and $-b \in \texttt{prem}(B^\perp)$, so $B^\perp$ is inconsistent with $B^\perp$. In other words, l is inconsistently attacked in AS.

To conclude, in all cases, l is inconsistently supported or attacked in AS.

Blocked Suppose that for each $AS' \in F(AS)$, l is blocked in AS' and $L'(l) \neq \langle 0,0,0,1\rangle$. We consider both the case that l is labelled $u(l) \vee o(l)$ and the alternative case that l is labelled $\neg u(l) \wedge \neg o(l)$ by L'.

- First suppose that l is labelled $u(l) \vee o(l)$ by L'. Since l is blocked in AS' for each $AS' \in F(AS)$, we know that for each $AS' \in F(AS)$, l is not unsatisfiable or out in AS'. Then by Lemma 17, l is inconsistently supported in AS.
- Alternatively, l is labelled $\neg u(l) \wedge \neg o(l)$ by L' but still $L' \neq \langle 0,0,0,1\rangle$. Then l should be labelled $d(l) \wedge b(l)$ (if $\neg b(l)$ then soundness of the defended case, proven in Proposition 2, would be violated).

  So we have that l is labelled $d(l)$. By Lemma 16.4 there is a potential argument $A^p \in P(AS)$ for l such that each argument B in $Arg(AS)$ that p-attacks A^p is p-attacked by some observation-based potential argument $C^p \in P(AS)$. Let A^p be an arbitrary potential argument for l in $P(AS)$ with these properties. So each argument B in $Arg(AS)$ that p-attacks A^p is p-attacked by some observation-based potential argument $C^p \in P(AS)$.

  Next, we show that each **potential** argument $B^p \in P(AS)$ that p-attacks A^p is p-attacked by some observation-based potential argument $C^p \in P(AS)$: suppose that there is some $B^p \in P(AS)$ that p-attacks A^p and is **not** attacked by any observation-based potential argument in $P(AS)$. Then $B^p \notin Arg(AS)$. So there is a $p \in \texttt{prem}(B^p)$ such that $p \in \mathcal{Q}$ and $p \notin \mathcal{K}$. Then there is an observation-based potential argument $-p$ in $P(AS)$, which p-attacks B^p; contradiction. So each potential argument $B^p \in P(AS)$ that p-attacks A^p is p-attacked by some observation-based potential argument $C^p \in P(AS)$.

  Now construct the set consisting of all observation-based potential arguments that attack potential arguments that attack A^p:

$$\mathcal{K}^{Def} = \{-b \mid B^p \in P(AS),\ B^p \text{ p-attacks } A^p,\ B' \in \texttt{sub}(B^p), \texttt{conc}(B') = b \\ b \in \mathcal{Q},\ b \notin \mathcal{K},\ -b \notin \mathcal{K}\}$$

  Next, we will prove by contradiction that l is inconsistently attacked in AS. Suppose that l is not inconsistently attacked in AS.

$\mathcal{K}^{Def}$ could be either consistent or inconsistent. If $\mathcal{K}^{Def}$ is inconsistent, then there is some B^p, C^p in $P(AS)$ such that both B^p and C^p p-attack A^p and B^p is inconsistent with C^p. A^p is a potential argument for l in $P(AS)$, so by Definition 12, l would be inconsistently attacked in AS. This contradicts our assumption, so we derive that $\mathcal{K}^{Def}$ is consistent.

We next show by contradiction that $\texttt{prem}(A^p) \cup \mathcal{K}^{Def}$ is consistent. Suppose that $\texttt{prem}(A^p) \cup \mathcal{K}^{Def}$ is inconsistent. Then there is some $p \in \texttt{prem}(A^p)$ such that $-p \in \mathcal{K}^{Def}$. By definition of $\mathcal{K}^{Def}$, there must be some potential argument B^p attacking A^p and there is some $B' \in \texttt{sub}(B^p)$ with $p \in \texttt{prem}(B')$, $p \in \mathcal{Q}$, $p \notin \mathcal{K}$ and $-p \notin \mathcal{K}$. Given that $-p \in \mathcal{Q}$ and $p \notin \mathcal{K}$ there is an observation-based potential argument $C^p = -p \in P(AS)$. $p \in \texttt{prem}(A^p)$, so C^p p-attacks A^p. Then $p \in \mathcal{K}^{Def}$. Since both p and $-p$ are in $\mathcal{K}^{Def}$, $\mathcal{K}^{Def}$ is inconsistent; contradiction. So $\texttt{prem}(A^p) \cup \mathcal{K}^{Def}$ is consistent.

Furthermore, $\mathcal{K} \subseteq \mathcal{K} \cup \mathcal{K}^{Def} \cup \texttt{prem}(A^p) \subseteq \mathcal{Q}$. So $AS^{Def} = (\mathcal{L}, \mathcal{R}, \mathcal{Q}, \mathcal{K} \cup \mathcal{K}^{Def} \cup \texttt{prem}(A^p))$ must be an argumentation setup in $F(AS)$.

Then l is defended in AS^{Def}: since $\texttt{prem}(A^p) \subseteq \mathcal{K} \cup \mathcal{K}^{Def} \cup \texttt{prem}(A^p)$, we have that $A^p \in Arg(AS^{Def})$. Let B an arbitrary argument in $Arg(AS^{Def})$ that attacks A^p. If $B \in Arg(AS^{Def})$, then $B \in P(AS)$. But we concluded earlier that each potential argument $B^p \in P(AS)$ that p-attacks A^p is p-attacked by some observation-based potential argument $C^p \in P(AS)$. Thanks to the way we defined $\mathcal{K}^{Def}$, each premise of C^p is in $\mathcal{K}^{Def}$, so $C^p \in Arg(AS^{Def})$. Since we picked B as an arbitrary attacker of A^p in $Arg(AS^{Def})$, we have that each argument attacking A^p in $Arg(AS^{Def})$ is attacked by an observation-based argument C^p in $Arg(AS^{Def})$. Then by Lemma 3, A^p is in the grounded extension $G(AS^{Def})$. Then by Definition 6, l is defended in AS^{Def}, which (together with our earlier conclusion that $AS^{Def} \in F(AS)$) contradicts our assumption that for each $AS' \in F(AS)$, l is blocked in AS'.

So we have to retract our assumption that l is not inconsistently attacked in AS. In other words, l is inconsistently attacked in AS.

To conclude, in all cases that l is stable in AS but l is not labelled stable by L', l is inconsistently supported or attacked in AS. □

3.4.3 Time complexity

Finally, the proposed algorithm runs in polynomial time, which makes it suitable for practical applications such as human-computer inquiry dialogues.

Proposition 4 (Time complexity stability labelling)**.** *The time complexity of* STABILITY *is* $\mathcal{O}(|\mathcal{L}|^2 \cdot |\mathcal{R}| + |\mathcal{L}| \cdot |\mathcal{R}|^2)$.

Proof. As shown in Lemma 7, the time complexity of the preprocessing step is $\mathcal{O}(|\mathcal{L}| \cdot |\mathcal{R}|^2)$. In Lemma 6, we show that the time complexity of the labelling step is $\mathcal{O}(|\mathcal{L}|^2 \cdot |\mathcal{R}| + |\mathcal{R}|^2)$ time. To conclude, the total time complexity of the STABILITY algorithm is $\mathcal{O}(|\mathcal{L}|^2 \cdot |\mathcal{R}| + |\mathcal{L}| \cdot |\mathcal{R}|^2)$. □

References

[1] Phan Minh Dung. On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games. Artificial Intelligence, 77:321–357, 1995.

[2] Daphne Odekerken, AnneMarie Borg, and Floris Bex. Estimating stability for efficient argument-based inquiry. In Computational Models of Argument: Proceedings of COMMA 2020, 2020. under review.

[3] Henry Prakken. An abstract framework for argumentation with structured arguments. Argument & Computation, 1(2):93–124, 2010.

[4] Bas Testerink, Daphne Odekerken, and Floris Bex. AI-assisted message processing for the Netherlands National Police. In Proceedings of the 1st Workshop on Artificial Intelligence and the Administrative State, 2019.

[5] Bas Testerink, Daphne Odekerken, and Floris Bex. A method for efficient argument-based inquiry. In Proceedings of the 13th International Conference on Flexible Query Answering Systems, 2019.