# Topological Conditions and Solutions for Repairing Argumentation Frameworks [⋆]

Kazuko Takahashi and Hiroyoshi Miwa

Kwansei Gakuin University, 1 Gakuen Uegahara, Sanda, 669-1330, JAPAN
`ktaka@kwansei.ac.jp, miwa@kwansei.ac.jp`

**Abstract.** This paper discusses how to make an argumentation framework (AF) with no stable extensions into one with a stable extension by adding a new argument, which we call 'repair'. We remove the restrictions that were put on the target AFs in our previous work, and show a simple condition for an arbitrary AF to have no stable extensions. Then, we refine the conditions that an AF should satisfy to be repaired and identify the position where a new argument is added. We also discuss other possible repair types. The judgments are simple, easy to intuitively understand by virtue of the usage of topological features.

**Keywords:** abstract argumentation framework, computational argumentation, dynamic argumentation, graph topology

## 1 Introduction

Dung's abstract Argumentation Framework (AF) is a standard model that formalizes argumentations [19]. It is a powerful tool for handling conflict, and has been applied in various research areas in the field of artificial intelligence, including decision making, non-monotonic reasoning, and agent communication. In an abstract AF, an argumentation is represented as a directed graph ignoring the contents of arguments and focused on the structure of the argumentation. Many extended frameworks for the AF and new semantics have been proposed so far [1, 2, 12, 21].

When an odd number of arguments constitute a cycle, the entire argumentation becomes stuck and no outcome is obtained. This may occur in an actual argumentation, and the state can be resolved by providing a counter-argument to a suitable argument.

In semantics of AFs, an AF including an odd-length cycle may not have a stable extension. Several semantics, e.g., CF2 [3], have been introduced to solve this problem. However, stable semantics most closely reflects the situation in the actual argumentation in which all the attendants agree to the accepted arguments and to the rejection of the other arguments. Particularly, in making a crucial decision such as a legal judgment on a trial or a policy of medical treatment of a patient on a tumor board, it is strongly required that at least

[⋆] This work was supported by JSPS KAKENHI Grant Number JP17H06103.

one such outcome is obtained that all the attendants agree to accept, and that overcomes the other counter-arguments. Stable semantics is most suitable to treat such cases.

In this paper, we investigate the case when an AF does not have a stable extension, and how to obtain an AF with a stable extension by adding a new argument to an appropriate position, which we call *repair*. However, it is difficult to quickly identify the position when the AF is large. For example, the AF shown in Figure 1 has no stable extensions. If we add a new argument attacking the argument $C$, then the AF is changed into the one with a stable extension. So far, a necessary and sufficient condition for the existence of a stable extension was shown [23, 11]. However, in these studies, the stability is judged using a certain semantics different from a stable one (for example, a preferred extension), which means that such an extension should be detected first. It would be desirable to find a condition without considering the other semantics.

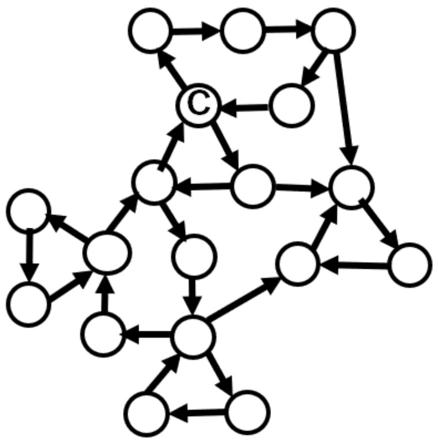


**Fig. 1.** AF with no stable extensions.

In this paper, we show the condition for a given AF not having a stable extension, and identify the position where a new argument is added to the reduced AF using its topological feature.

Previously, we investigated a simple AF consisting of connected cycles and the length of each cycle is three [25], and then extended our target AFs to those that allow general odd-length cycles and proposed a reduction approach [24]. A given AF is shrunk to a simple form and its stability and repairability are discussed. However, the target AFs were still restricted.

In this paper, we treat an arbitrary AF by removing all of these restrictions, and refine the reduction procedure. First, we clarify the topological condition of AF for not having a stable extension, which covers a wide range. Next, we discuss the repair of a reduced AF. We describe the topological condition for repairability. Then, we identify the positions where new arguments are added in several repair types.

This paper is organized as follows. In Section 2, we describe basic concepts. In Section 3, we formalize a reduction procedure. In Section 4, we show the condition for an unstable AF. In Section 5, we discuss repair of the reduced AFs. In Section 6, we compare our approach with related works. Finally, in Section 7, we present our conclusions and directions for future research.

## 2 Preliminaries

The abstract AF proposed by Dung [19] is a representation of an argumentation structure that ignores its content.

**Definition 1 (argumentation framework)** *An* argumentation framework (AF) *is defined as a pair $\langle \mathcal{A}, \mathcal{R} \rangle$ where $\mathcal{A}$ is a set of arguments and $\mathcal{R} \subseteq \mathcal{A} \times \mathcal{A}$.*

A pair $(A, B) \in \mathcal{R}$ is called *an attack*, and it is said that $A$ *attacks* $B$.

An AF can be represented as a directed graph in which each node corresponds to an argument, and each edge corresponds to an attack. In this paper, we consider a finite AF that can be represented as a connected finite directed graph.

**Definition 2 (path,cycle)** *Let $\langle \mathcal{A}, \mathcal{R} \rangle$ be an AF and $A_0, A_n \in \mathcal{A}$. If there exists a sequence of attacks $(A_0, A_1), (A_1, A_2), \ldots, (A_{n-1}, A_n) \in \mathcal{R}$ where for all $i, j$; $0 \leq i \neq j \leq n-1$, $A_i \neq A_j$, then $\langle A_0, \ldots, A_n \rangle$ is called* a path from $A_0$ to $A_n$, *and $n$ is called its* length. *For a path $\langle A_0, \ldots, A_n \rangle$, if $A_n = A_0$ then it is called* a cycle from $A_0$ to $A_0$.

*Example 1.* In an AF in Figure 2, for example, $\langle a, b, c \rangle$ is a path of length 2, $\langle a, b, e, a \rangle$ is a cycle of length 3, $\langle a, a \rangle$ is a cycle of length 1, but $\langle a, b, c, d, b, e, a \rangle$ is not a cycle.

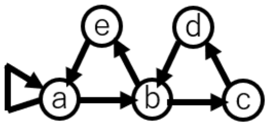


**Fig. 2.** Example of an AF.

For an abstract AF, semantics is defined either by an extension or labeling, which have a one-to-one relation with each other [1]. In this paper, we adapt semantics by labeling.

Labeling is a total function from a set of arguments to a set of labels $\{in, out, undec\}$.

**Definition 3 (complete labeling)** *Let $\langle \mathcal{A}, \mathcal{R} \rangle$ be an AF. A labeling $\mathcal{L}$ is called* a complete labeling *if the following conditions are satisfied for any argument $A \in \mathcal{A}$.*

- $\mathcal{L}(A) = in$ *iff* $\forall B \in \mathcal{A}; (B, A) \in \mathcal{R} \Rightarrow \mathcal{L}(B) = out$.
- $\mathcal{L}(A) = out$ *iff* $\exists B \in \mathcal{A}; \mathcal{L}(B) = in \wedge (B, A) \in \mathcal{R}$.
- $\mathcal{L}(A) = undec$*, otherwise.*

Hereafter, the term “labeling” denotes complete labeling unless otherwise indicated. The set $\{A | A \in \mathcal{A}, \mathcal{L}(A) = in\}$ is a set of accepted arguments corresponding to an extension in the extension-based semantics.

**Definition 4 (stable labeling)** *Let $\langle \mathcal{A}, \mathcal{R} \rangle$ be an AF. For a complete labeling $\mathcal{L}$, if $\{A | A \in \mathcal{A}, \mathcal{L}(A) = undec\} = \emptyset$, then it is called* a stable labeling.

There exists an AF that has no stable labelings.

**Definition 5 (stable/unstable AF)** *An AF with a stable labeling is called* a stable AF*, and one without it is called* an unstable AF.

In addition to these concepts, we introduce several new concepts and terminologies.

**Definition 6 (connector)** *Let $\mathcal{F} = \langle \mathcal{A}, \mathcal{R} \rangle$ be an AF. For an argument $B \in \mathcal{A}$, if there exists more than one argument $A$ such that $(A, B) \in \mathcal{R}$, then $B$ is said to be* a connector of $\mathcal{F}$*; if there exists a unique argument $A$ such that $(A, B) \in \mathcal{R}$, then $B$ is said to be* a non-connector of $\mathcal{F}$.

Note that an argument without an attack is neither a connector nor a non-connector. For brevity, we call an attack from a non-connector *an nc-attack*.

**Definition 7 (nc-cycle)** *Let $\mathcal{F}$ be an AF. A cycle $\langle A_0, \ldots, A_{n-1}, A_0 \rangle$ in $\mathcal{F}$ where all $A_i$ $(0 \leq i \leq n-1)$ are non-connectors is said to be* a nc-cycle of $\mathcal{F}$.

**Definition 8 (cpath)** *Let $C, D$ be (possibly the same) connectors of an AF. Then, the path $\langle C, A_1, \ldots, A_n, D \rangle$ where $A_1, \ldots, A_n$ $(n \geq 1)$ are non-connectors is said to be* a cpath from $C$ to $D$.

Note that there exists an nc-cycle with length 1, i.e., a self-attack, whereas the length of any cpath is more than 1.

*Example 2.* In the AF in Figure 3, $d$ and $f$ are the connectors, a cycle $\langle a, b, c, a \rangle$ is the nc-cycle, paths $\langle d, e, f \rangle$, $\langle d, e, h, d \rangle$ and $\langle f, g, d \rangle$ are the cpaths.

**Definition 9 (annihilator,entrance)** *Let $\mathcal{F} = \langle \mathcal{A}, \mathcal{R} \rangle$ be an AF. A pair of an argument $A \notin \mathcal{A}$ and an attack $(A, B)$ to $B \in \mathcal{A}$ is said to be* an annihilator of $\mathcal{F}$*, and $B$ is said to be an* entrance of $\mathcal{F}$.

We revise an unstable AF by adding annihilators to obtain a stable AF.

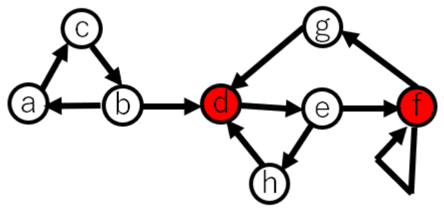


**Fig. 3.** Connector, nc-cycle, and cpath.

**Definition 10 ($k$-repair)** *Let $\langle \mathcal{A}, \mathcal{R} \rangle$ be an unstable AF. For $A_1, \ldots, A_k \notin \mathcal{A}$ where $A_i \neq A_j$ for any $i, j$ $(1 \leq i \neq j \leq k)$, set $\mathcal{A}' = \mathcal{A} \cup \{A_1, \ldots, A_k\}$, and also set $\mathcal{R}' = \{(A_1, B_1), \ldots, (A_k, B_k)\}$ where for all $i$ $(1 \leq i \leq k)$, $B_i \in \mathcal{A}$, and for all $i, j$ $(1 \leq i \neq j \leq k)$ $B_i \neq B_j$. Then the act of revision from $\langle \mathcal{A}, \mathcal{R} \rangle$ to $\langle \mathcal{A}', \mathcal{R}' \rangle$ is said to be* a $k$-repair.

Hereafter, in figures, a red node denotes a connector, and in the figures showing a labeling, a pink node denotes an argument labeled *in* and a blue node denotes an argument labeled *out*; a rectangle with an arrow denotes an annihilator.

*Example 3.* The AF shown in Figure 2 is unstable. Figure 4 shows the result of 1-repair by adding an annihilator to an argument $b$. This AF is stable (Figure 4).

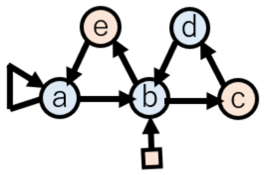


**Fig. 4.** Result of 1-repair of an AF shown in Figure 2.

## 3 Reduction

It is difficult to understand the structure of a large and complicated AF, and it is computationally intensive to explore its stability or repairability directly. We introduced the reduced form of a given AF, preserving the labels of connectors, as the label of the connector is the key to considering stability [24].

The reduction procedure contains shrinkage of nc-cycles and that of paths. As a pair of succeeding non-connectors in a path to a connector do not affect the label of the connector in the path, we shrink a subsequent pair of non-connectors for each path. In addition, an nc-cycle $q$ is shrunk to one special node called an *'undec-node'* with a self-attack, denoted by $\mathcal{U}_q$. At the same time, the attacks from a node in the nc-cycle to its outside node are reconnected to the attacks from $\mathcal{U}_q$.

**[Reduction procedure]**[1].

Let $\mathcal{F} = \langle \mathcal{A}, \mathcal{R} \rangle$ be a given AF. Repeat the following procedure as far as possible.

1. (shrink nc-cycles)
   (a) For each nc-cycle $q = \langle A_{q1}, A_{q2}, \ldots, A_{qm_q}, A_{q1} \rangle$ in $\mathcal{F}$, we define four sets:
      – $F_q = \{A_{q1}, A_{q2}, \ldots, A_{qm_q}\}$,
      – $G_q = \{(A_{q1}, A_{q2}), \ldots, (A_{qm_q}, A_{q1})\}$,
      – $H_q = \{(A, X) | (A, X) \in \mathcal{R}, A \in q, X \notin q\}$,
      – $J_q = \{(\mathcal{U}_q, \mathcal{U}_q)\} \cup \{(\mathcal{U}_q, X) | (A, X) \in H_q\}$.
   (b) Set $F = \bigcup_q F_q$, $G = \bigcup_q G_q$, $H = \bigcup_q H_q$, $J = \bigcup_q J_q$ and $U = \bigcup_q \{\mathcal{U}_q\}$.
   (c) Set $\mathcal{A}_0 = (\mathcal{A} \setminus F) \cup U$, $\mathcal{R}_0 = (\mathcal{R} \setminus (G \cup H)) \cup J$. Then, we get $\mathcal{F}_0 = \langle \mathcal{A}_0, \mathcal{R}_0 \rangle$.
2. (shrink paths)
   (a) For each path $p = \langle C_p, A_{p1}, A_{p2}, \ldots, A_{pk_p}, D_p \rangle$ $(k_p > 1)$ in $\mathcal{F}_0$, where $C_p$ is a connector or an undec-node, $A_{p1}, A_{p2}, \ldots, A_{pk_p}$ are non-connectors, and $D_p$ is a connector, we define three sets:
      – $S_p = \{A_{p1}, \ldots, A_{pk_p}\}$,
      – $T_p = \{(C_p, A_{p1}), (A_{p1}, A_{p2}), \ldots, (A_{pk_p}, D_p)\}$,
      – $V_p = \{(C_p, D_p)\}$ if $k_p$ is even, $V_p = \{(C_p, E_p), (E_p, D_p)\}$ where $E_p$ is a new argument if $k_p$ is odd.
   (b) Set $S = \bigcup_p S_p$, $T = \bigcup_p T_p$, $E = \bigcup_p E_p$ and $V = \bigcup_p V_p$.
   (c) Set $\mathcal{A}_1 = (\mathcal{A}_0 \setminus S) \cup E$, $\mathcal{R}_1 = (\mathcal{R}_0 \setminus T) \cup V$. Then we get $\mathcal{F}_1 = \langle \mathcal{A}_1, \mathcal{R}_1 \rangle$.
3. Set $\mathcal{F} = \mathcal{F}_1$.

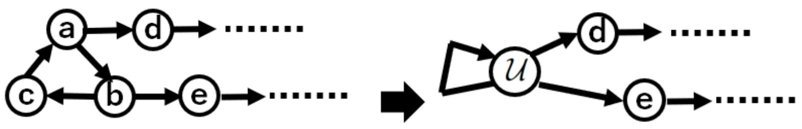
*Example 4.* Figure 5 shows an example of shrinkage of an nc-cycle. The nc-cycle $\langle a, b, c, a \rangle$ in Figure 5(a) is shrunk to an undec-node $\mathcal{U}$; the attacks $(a, d)$ and $(b, e)$ are replaced by $(\mathcal{U}, d)$ and $(\mathcal{U}, e)$, respectively (Figure 5(b)).

The reduction procedure terminates because AF is finite, and the number of connectors never increases.

**Definition 11 (reduced form of AF)** *Let $\mathcal{F}$ be an AF. The AF finally obtained by the reduction procedure is said to be* a reduced form of $\mathcal{F}$.

*Example 5.* Figure 6 shows an example of reduction. Figure 6(a) is a given AF. The cpath from $D$ to $C$ contains the subsequent non-connectors that are deleted and the path is shrunk to a direct edge from $D$ to $C$. Two cpaths from $C$

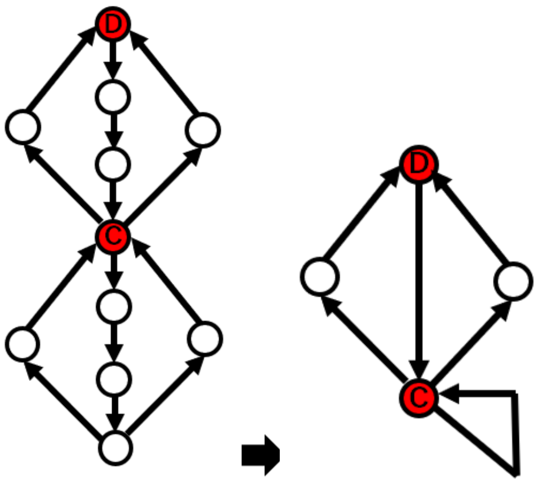
[1] The definition of the reduction is modified from that described in [24].



(a) nc-cycle (b) shrunk form

**Fig. 5.** Shrinkage of nc-cycles.

to $D$ do not change since both contain only one intermediate node between the connectors, respectively. Both of the two cpaths from $C$ to $C$ have four intermediate non-connectors, respectively, therefore, these paths are shrunk to self-attacks of $C$, which are merged to the single self-attack. As a result, we have the reduced AF shown in Figure 6(b). In this case, both connectors of $\mathcal{F}$ remain as connectors in the reduced AF.



(a) Given AF (b) Reduced AF

**Fig. 6.** Reduction: both connectors remain.

*Example 6.* Figure 7 shows another example of reduction. There are two cpaths from $C$ to $D$ (Figure 7(a)), both of which are shrunk and merged to the single edge from $C$ to $D$. Similarly, two cpaths from $C$ to $C$ are shrunk and merged to the single self-attack of $C$ (Figure 7(b)). As a result, $D$ is no longer a connector, and there appears a new cpath $\langle C, a, b, C\rangle$ from $C$ to $C$ (Figure 7(b)). Then, repeat the procedure. This cpath is shrunk to a self-attack and merged with the existing self-attack, and we obtain the single node with the self-attack which is no more a connector (Figure 7(c)). Then, repeat the procedure again. This nc-cycle is reduced to $\mathcal{U}$ (Figure 7(d)). Finally, the reduced AF consists of only one undec-node with a self-attack. In this case, both connectors of $\mathcal{F}$ disappear in the reduced form.

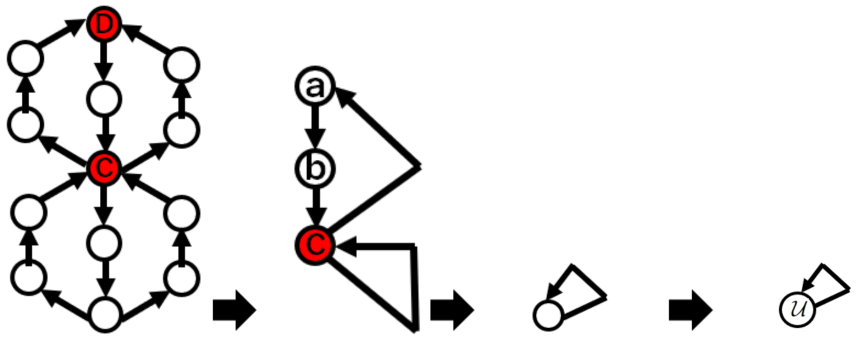


**Fig. 7.** Reduction: both connectors disappear.

The nodes in the reduced form are classified into three types: connector, non-connector, and undec-node depending on the number of their attackers.

The reduced AF has the following properties.

**Proposition 1** *1. Each cpath (in the reduced AF) includes exactly one non-connector.*

*2. Each path from an undec-node to a connector in which no connector appears includes at most one non-connector.*

*3. An undec-node has no attacker except for itself.*

*Proof.* 1. The length of each cpath in the original AF is more than one, and subsequent non-connectors in the cpath are deleted by a pair at the step 2 in the reduction procedure. Therefore, the number of the remaining non-connectors in a cpath is one.

2. Let $p$ be a path from an undec-node to a connector in which no connector appears in the original AF. Subsequent non-connectors in $p$ are deleted by a pair at the step 2 in the reduction procedure. Therefore, the number of the remaining non-connectors in $p$ is at most one.

3. An undec-node is added with a self-attack only at the step 1(c), and no other attacks are added to it.

## 4 Judgment for Unstability

It is possible to easily judge unstability only by checking its topology if an AF has some topological property. We first show the class of such AFs and how to repair them in the next section.

**Definition 12 (alternate-io-path)** *Let $\mathcal{F}$ be an AF with a stable labeling $\mathcal{L}$. A path $p = \langle A_0, \ldots, A_n \rangle$ $(n > 0)$, in which $\mathcal{L}(A_0), \ldots, \mathcal{L}(A_n)$ are assigned in and out in turn is said to be* an alternate-io-path w.r.t. $\mathcal{L}$.

**Theorem 1 (unstability of AF)** *If an AF satisfies the following two conditions, then it is unstable.*
**[COND1]**

1. *There exists no even-length cycle.*
2. *Each argument is attacked by at least one argument (including itself).*

*Proof.* Assume that $\mathcal{F} = \langle \mathcal{A}, \mathcal{R} \rangle$ has a stable labeling $\mathcal{L}$.

For an arbitrary argument $A \in \mathcal{A}$, let $p$ be the longest alternate-io-path w.r.t. $\mathcal{L}$ which starts from $A$. Then, there exists an argument $B \in \mathcal{A}$ that attacks $A$ from the second condition.

We show that contradiction occurs, by splitting cases.
(1) $B \notin p$.

(1.1) If $\mathcal{L}(A) = in$, then $\mathcal{L}(B) = out$. It follows that there exists an alternate-io-path longer than $p$; it is a contradiction.

(1.2) If $\mathcal{L}(A) = out$ and $\mathcal{L}(B) = in$, then there also exists an alternate-io-path longer than $p$; it is a contradiction.

(1.3) If $\mathcal{L}(A) = out$ and $\mathcal{L}(B) = out$, then there should be an argument $C \in \mathcal{A}$ that attacks $A$ and $\mathcal{L}(C) = in$. If $C \notin p$, then there exists an alternate-io-path longer than $p$; it is a contradiction. If $C \in p$, there exists a cycle from $A$ to $A$ that consists of the alternate-io-path from $A$ to $C$ followed by an attack $(C, A)$. It is a cyclic alternate-io-path of which the length is even; which contradicts the first condition.
(2) $B \in p$

(2.1) If $\mathcal{L}(A) = in$, then $\mathcal{L}(B) = out$, there exists a cycle from $A$ to $A$ that consists of the alternate-io-path from $A$ to $B$ followed by an attack $(B, A)$. It is a cyclic alternate-io-path of which the length is even; which contradicts the first condition.

(2.2) If $\mathcal{L}(A) = out$ and $\mathcal{L}(B) = in$, contradiction by the same reason with the case (2.1).

(2.3) If $\mathcal{L}(A) = out$ and $\mathcal{L}(B) = out$, then there should be a non-connector $C \in \mathcal{A}$ that attacks $A$. If $C \notin p$, then there exists an alternate-io-path longer than $p$; it is a contradiction. If $C \in p$, there exists a cycle from $A$ to $A$ that consists of the alternate-io-path from $A$ to $C$ followed by an attack $(C, A)$. It is a cyclic alternate-io-path of which the length is even; which contradicts the first condition.

Therefore, contradiction occurs in all cases. Thus, $\mathcal{F}$ is unstable. □

**Corollary 1** *Let $\mathcal{F}$ be an AF and $\mathcal{F}'$ be its reduced form. If $\mathcal{F}'$ satisfies [COND1], then $\mathcal{F}$ is unstable.*

*Proof.* $\mathcal{F}$ includes no even-length cycle if and only if $\mathcal{F}'$ includes no even-length cycle, and each argument is attacked by at least one argument in $\mathcal{F}$ if and only if each argument is attacked by at least one argument in $\mathcal{F}'$, since a subsequent non-connectors are deleted by a pair in the reduction procedure. Therefore, $\mathcal{F}$ satisfies [COND1] if and only if $\mathcal{F}'$ satisfies [COND1]. Therefore, if $\mathcal{F}'$ satisfies [COND1], then $\mathcal{F}$ is unstable. $\square$

This result shows that we can judge unstability of an AF by checking the topology of the reduced AF, and we can also discuss repairability on the reduced AF, assuming [COND1].

## 5 Repair of Reduced AF

### 5.1 1-repair with out-labeled connector

Next, we show how to identify an entrance on 1-repair on the reduced AF.

An undec-node is an argument without an nc-attack. Therefore, we treat undec-node and connectors without nc-attacks alike when identifying an entrance.

For a reduced AF $\mathcal{F}'$, we denote $\mathcal{C}_{\mathcal{F}'}$ a set of connectors without nc-attacks and undec-nodes of $\mathcal{F}'$.

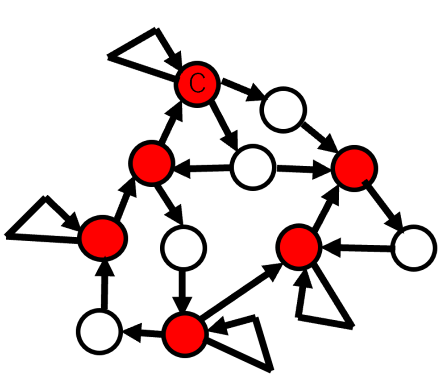
**Theorem 2 (1-repairability of reduced AF)** *Let $\mathcal{F}'$ be a reduced AF that satisfies [COND1]. If $\mathcal{C}_{\mathcal{F}'} = \{C\}$, then it is 1-repairable by taking $C$ as an entrance, and each connector is labeled out in the repaired AF.*

*Proof.* $\mathcal{F}'$ is unstable from Theorem 1. Let $\mathcal{L}$ be a labeling of the resulting AF that gives all non-connectors including an annihilator *in* and the others *out*. An annihilator is labeled *in* since it has no attacker. Each connector including $C$ has an nc-attack in the resulting AF, and thus it is labeled *out*. Each non-connector is labeled *in* since it is attacked only by the connector which is labeled *out*. Therefore, $\mathcal{L}$ is stable, and each connector is labeled *out* in the repaired AF. $\square$

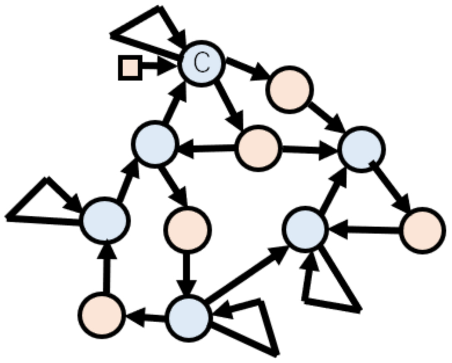
*Example 7.* Figure 8(a) is the reduced form of the AF shown in Figure 1. In this AF, the node $C$ is the only connector that has no nc-attack, which is identified as an entrance and so the resulting AF is stable (Figure 8(b)).

The condition shown in this theorem provides a simple intuitional method for repair: checking the attacks of each connector. This matches the definition of stable extension, i.e., the set of arguments labeled *in* is conflict-free and attacks all the arguments outside of the set. The computational complexity of the judgment of 1-repairability and identification of an entrance is linear.

We can derive the following theorem regarding this type of repair from the propositions shown in [24].



(a) The reduced AF



(b) Labeling to the repaired AF

**Fig. 8.** Repair of the reduced AF.

**Theorem 3 (1-repairability of AF)** *Let $\mathcal{F}$ be an AF without an nc-cycle, and $\mathcal{F}'$ be its reduced form that satisfies [COND1].*
*(1) $\mathcal{F}$ is 1-repairable if and only if $\mathcal{F}'$ is 1-repairable.*
*(2) When all the connectors in $\mathcal{F}$ remain in $\mathcal{F}'$ and if $\mathcal{F}'$ is 1-repairable by taking an argument $E$ as an entrance, then $\mathcal{F}$ is 1-repairable by taking $E$ as an entrance.*

### 5.2 $k$-repair with out-labeled connector

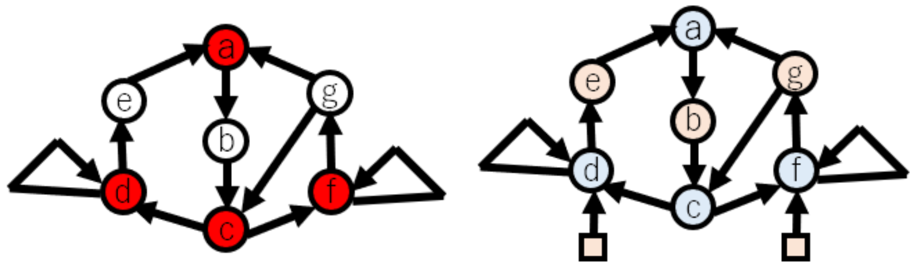
Next, we discuss $k$-repair.

**Theorem 4 (k-repairability of reduced AF)** *Let $\mathcal{F}'$ be a reduced AF that satisfies [COND1]. If $|\mathcal{C}_{\mathcal{F}'}| = k$, then it is $k$-repairable by taking all the arguments in $\mathcal{C}_{\mathcal{F}'}$ as entrances, and each connector is labeled out in the repaired AF.*

*Proof.* $\mathcal{F}'$ is unstable from Theorem 1. If we add annihilators to all the arguments in $\mathcal{C}_{\mathcal{F}'}$, all the entrances are labeled *out*, and the resulting AF has a stable labeling by which all the connectors are labeled *out* and all the non-connectors are labeled *in*, by the same reason with that of Theorem 2. □

*Example 8.* The reduced AF shown in Figure 9(a) has two connectors $d$ and $f$ that have no nc-attacks. It is repaired by taking these two connectors as entrances (Figure 9(b)). It is 2-repair.

We may have another result of $k$-repair where $k$ is less than the number of the arguments in $\mathcal{C}_{\mathcal{F}'}$. In this case, some connectors are labeled *in*. We will show an example in the next subsection.



(a) The reduced AF (b) Labeling to the $k$-repaired AF

**Fig. 9.** Example of 2-repair.

### 5.3 1-repair with in-labeled connector

All the connectors are labeled *out* in the repaired AF by the type of repair mentioned in subsections 5.1 and 5.2. Then, can we make a connector labeled *in*? And if possible, where is an entrance? Are there any topological constraints on an AF? These are the next issues to be discussed.

A connector can be considered as an argument corresponding to one of significant claims in the entire argumentation, since it is attacked by several arguments. Therefore, it is meaningful to make a connector to be labeled *in*, that is, accepted. Such a repair gives a strategy to persuade the other agents to accept an agent's main claim.

We can make a specific connector labeled *in* if all its attackers are labeled *out*. But to realize it by adding only one annihilator, the entrance should be taken in the shared part of all the paths in which these attackers are, respectively. Moreover, an additional condition is required. If there exists a subsequent connectors in a path from the entrance to the attacker of the specified connector, the former connector should have a self-attack. Since a connector with a self-attack cannot be labeled *in* by any labeling, it is labeled *out*. Therefore, the latter connector cannot be labeled *out*. It may cause a conflict. Not all AFs can avoid this conflict, but AFs with some topology can. In the followings we show two topological conditions on this type of repairability.

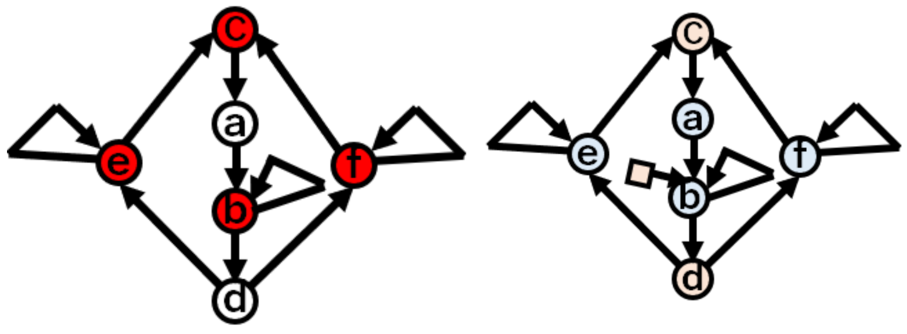
**[COND2]**
Let $\mathcal{F}'$ be a reduced AF that satisfies [COND1].

1. $\mathcal{C}_{\mathcal{F}'} = \{C\}$.
2. $C$ does not have a self-attack.
3. The cycles included in $\mathcal{F}'$ are only those from $C$ to $C$, all of which share the path $\langle C, A_1, A_2 \rangle$.
4. In each cycle $\langle C, A_1, A_2, A_3, \ldots, A_k, C \rangle$, $A_j$ is a non-connector if $j$ is odd and a connector if $j$ is even $(3 \leq j \leq k)$. (The length of the shared path may be more than two.)

**Proposition 2** *Let $\mathcal{F}'$ be a reduced AF that satisfies [COND1]. If $\mathcal{F}'$ satisfies [COND2], then it is 1-repairable by taking $A_2$ as an entrance, in this case $C$ is labeled in, in the repaired AF.*

*Proof.* $\mathcal{F}'$ is unstable from Theorem 1. Let $\mathcal{L}$ be a labeling to the resulting AF that for each path $\langle C, A_1, A_2, A_3, \ldots, A_k, C\rangle$, $\mathcal{L}(A_j) = in$ if $j$ is odd and $\mathcal{L}(A_j) = out$ if $j$ is even ($3 \leq j \leq k$). Then $\mathcal{L}(A_2) = out$, since $A_2$ is an entrance. For each path $\langle A_3, \ldots, A_k\rangle$, if $j$ is odd, $\mathcal{L}(A_j) = in$ should hold since $A_j$ is a non-connector which is attacked by only $A_{j-1}$ labeled *out*; if $j$ is even, $\mathcal{L}(A_j) = out$ should hold since $A_j$ is a connector which is attacked by $A_{j-1}$ labeled *in*. Then $\mathcal{L}(A_k) = out$, since $k$ is even. Therefore, $\mathcal{L}(C) = in$, since all the arguments that attack $C$ in all paths are labeled *out* and $C$ does not have a self-attack. Then $\mathcal{L}(A_1) = out$, since $A_1$ is attacked by $C$. It is consistent with the labeling $\mathcal{L}(A_2) = out$. Therefore, $\mathcal{L}$ is a stable labeling. □

*Example 9.* Figure 10(a) shows an AF that satisfies [COND2]. We get a stable AF by taking $b$ as an entrance; in this case, $C$ is labeled *in*, in the repaired AF (Figure 10(b)). It is 1-repair.



(a) An AF (b) Repaired AF

**Fig. 10.** Example of 1-repair with connector labeled *in*.

The next proposition shows another condition to get a stable AF with a labeling which gives *in* to multiple connectors by 1-repair.

We consider the case in which $\mathcal{C}_{\mathcal{F}'}$ has more than one argument and focus on cycles of one of these arguments. It is required that the entrance is shared by all the paths between the arguments in $\mathcal{C}_{\mathcal{F}'}$, and that all the attackers of the focused argument are labeled *out* in the repaired AF.

**[COND3]**

Let $\mathcal{F}'$ be a reduced AF that satisfies [COND1].

1. $\mathcal{C}_{\mathcal{F}'} = \{C^0, C^1, \ldots, C^n\}$ $(n > 0)$.
2. $C^0, C^1, \ldots, C^n$ do not have self-attacks.
3. The cycles appearing in $\mathcal{F}'$ are only those from $C^0$ to $C^0$, each of that does not include $C^i (\neq C^0) \in \mathcal{C}_{\mathcal{F}'}$.
4. All the paths from $C^0$ to $C^i$ $(0 \leq i \leq n)$ share the path $\langle C^0, A_1, A_2 \rangle$.
5. In each path $\langle C^0, A^i_1, A^i_2, A^i_3, \ldots, A^i_{k_i}, C^i \rangle$ from $C^0$ to $C^i$ $(0 \leq i \leq n)$, $k_i$ is even, $A^i_j$ is a non-connector if $j$ is odd and a connector if $j$ is even $(3 \leq j \leq k_i)$. (The length of the shared part may be more than two.)

**Lemma 1** *Let $\mathcal{F}'$ be a reduced AF that satisfies [COND1]. If $\mathcal{F}'$ satisfies [COND3], then for each $t$; $1 \leq t \leq n$, there is no path from $C^t$ to $C^i$ for each $i$ $(0 \leq i \leq n)$.*

*Proof.* $\mathcal{F}'$ is unstable from Theorem 1. For each $t$; $1 \leq t \leq n$, there exists a path from $C^0$ to $C^t$, since $C^t$ has an attack except for itself from the second condition.

Assume that there exists a path from $C^t$ to $C^0$. Then there exists a cycle from $C^0$ to $C^0$ including $C^t (\neq C^0) \in \mathcal{C}_{\mathcal{F}'}$, which contradicts the third condition. Therefore, there exists no path from $C^t$ to $C^0$.

Assume that there exists a path from $C^t$ to $C^i$ $(1 \leq i \neq t \leq n)$. If $A$ attacks $C^t$, then $A$ should be a connector from the first condition. It follows that $A$ and $C^t$ are succeeding connectors in the path $\langle C^0, A^i_1, A^i_2, A^i_3, \ldots, A^i_{k_i}, C^i \rangle$. Let $C^t = A^i_h$, then $h \geq 3$, since each cycle from $C^0$ to $C^0$ does not include $C^t$ from the fourth condition. If $h = 3$, then $C^t$ should be a non-connector from the fifth condition, which is a contradiction; if $h \geq 4$, then $A$ and $C^t$ are succeeding connectors, which contradicts the fifth condition.

Therefore, for each $t$; $1 \leq t \leq n$, there is no path from $C^t$ to $C^i$ for each $i$ $(0 \leq i \leq n)$. □

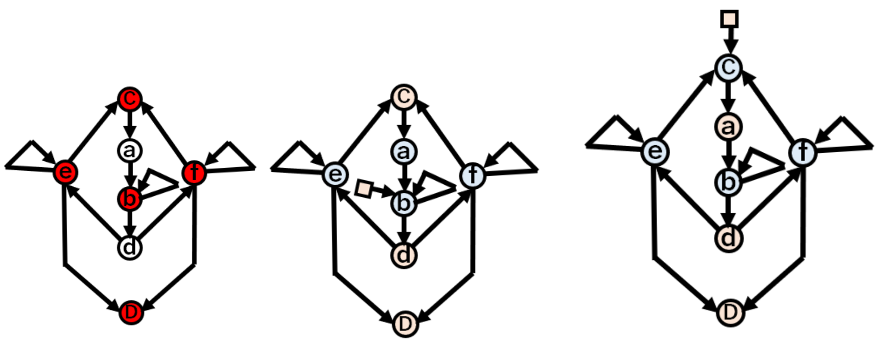
**Proposition 3** *Let $\mathcal{F}'$ be a reduced AF that satisfies [COND1]. If $\mathcal{F}'$ satisfies [COND3], then it is 1-repairable by taking $A_2$ or $C^0$ as an entrance. In both cases, $C^i$ is labeled in for all $i$ $(1 \leq i \leq n)$; and $C^0$ is labeled in, in the former case whereas labeled out, in the latter case, in the repaired AFs, respectively.*

*Proof.* $\mathcal{F}'$ is unstable from Theorem 1.
(1) $A_2$ is taken as an entrance.

Let $\mathcal{L}_1$ be a labeling to the resulting AF such that for each path $\langle C^0, A^i_1, A^i_2, A^i_3, \ldots, A^i_{k_i}, C^i \rangle$, $\mathcal{L}_1(A^i_j) = in$ if $j$ is odd and $\mathcal{L}_1(A^i_j) = out$ if $j$ is even $(3 \leq j \leq k_i)$. Then, $\mathcal{L}_1(A^i_{k_i}) = out$, since $k_i$ is even. From Lemma 1, there is no path from the argument $C^t \in \mathcal{C}_{\mathcal{F}'}$ $(t \neq 0)$ to $C^i$ $(0 \leq i \leq n)$. Therefore, $\mathcal{L}_1(C^i) = in$, since all the arguments that attack $C^i$ in all paths are labeled *out* and $C^i$ does not have a self-attack. Then $\mathcal{L}_1(A_1) = out$, since $A_1$ is attacked by $C^0$. It is consistent with the labeling $\mathcal{L}_1(A_2) = out$. Therefore, $\mathcal{L}_1$ is a stable labeling, which gives a label *in* to $C^0, C^1, \ldots, C^n$.
(2) $C^0$ is taken as an entrance.

Let $\mathcal{L}_2$ be a labeling to the resulting AF. Then $\mathcal{L}_2(C^0) = out$ since $C^0$ is an entrance. In this case, all the connectors in the cycles from $C^0$ to $C^0$ are labeled *out*, by the similar discussion with that in the case of (1). □

*Example 10.* Figure 11(a) shows an AF $\mathcal{F}'$ that satisfies [COND3]. $C$ and $D$ are two connectors without nc-attacks. That is, $\mathcal{C}_{\mathcal{F}'} = \{C, D\}$. We get a stable labeling $\mathcal{L}_1$ by taking $b$ as an entrance; in this case $\mathcal{L}_1(C) = in$ and $\mathcal{L}_1(D) = in$ (Figure 11(b)). We also get a stable labeling $\mathcal{L}_2$ by taking $C$ as an entrance; in this case $\mathcal{L}_2(C) = out$ and $\mathcal{L}_2(D) = in$ (Figure 11(c)).



(a) An AF (b) Repaired AF with $\mathcal{L}_1$ (c) Repaired AF with $\mathcal{L}_2$

**Fig. 11.** Example of 1-repair with connector labeled *in*.

## 6 Related Works

Dynamic argumentation is currently a focus of much research [18, 8]. Such studies evaluate changes in argumentation frameworks by adding/removing arguments/attacks, and mainly discuss changes in extensions caused by addition or removal operations. Earlier works have mainly investigated and compared changes in extensions in several extension-based semantics when the addition or removal of arguments/attacks are performed [13–16]. The problem which operations are required so that a desired set of arguments becomes a subset of an extension was introduced as an *enforcing problem* [6], and many studies have examined this [10, 5, 9, 17, 28].

The repair we discuss here can be considered as an enforcing problem. We focus on identifying a position by checking the topology of an AF, whereas most other studies on enforcing have focused on changes in extensions and have attempted to identify a minimal change by comparing solutions. For example, Baumann et al. considered a minimal change in AFs on the enforcing problem by introducing a value function of an AF based on a distance function between two AFs [7]. The 1-repair that we showed is considered a solution with a minimal change in a sense.

A necessary and sufficient condition for the existence of a stable extension are discussed in some works. Baumann et al. proved the condition for a given AF with an odd-length cycle [11]. The stability is judged using an admissible extension. They did not refer to the position to repair. Schulz et al. also investigated the condition [23]. They proposed two different approaches: labeling based one and structure based one. In both approaches, they characterized a given AF using preferred labeling, and showed the condition for stability. The crucial difference between these two works and ours is as follows: the (un)stability is judged regarding a certain semantics different from a stable one in their methods, which means that the result in the other semantics has to be obtained first; on the other hand, although it is not a necessary condition for stability, the unstability is judged just from a topological feature of an AF directly in our method, which means that if an AF has some specific topology, the judgment is done in a simple and quick manner. We have clarified the classes of AFs for which a repair can be found using topological properties.

Some works used the topological properties of an argument graph to treat dynamic argumentation frameworks [20, 4]. They used simple topological properties, such as symmetry and similarity, to reduce the complexity of computing changes in extensions, whereas we investigated the relationships among topological properties and the possibility of repair. Other works proposed a reduction of an AF using shrinking loops [26, 22] but did not discuss repair.

A repair shown in our work can be regarded as an abduction in logic programming in the sense of finding a minimal change in the knowledge base by adding a fact and a rule. Šefránek described the relationship between a dynamic argumentation framework and the revision of logic programming [27]. It would be interesting to relate our approach to an abduction of logic programming.

## 7 Conclusion

In this paper, we have discussed unstability and the repair of an AF using a reduction. We described the topological conditions of the reduced AF for repairability and identified the entrances. When an argumentation becomes stuck, we can easily find the position where a counter-argument should be added to lead to acceptance of an agent's claim. We have got more generally applicable results by removing the restrictions on the target AFs presented in our previous works.

Our main contributions are as follows:

– We have clarified a simple condition on the topological properties of an AF for its unstability, which covers a wide range.
– We have shown several ways of repair: 1-repair which makes the labels of all connectors *out*, 1-repair which makes the labels of some connectors *in* and $k$-repair.
– These judgments are simple, easy to understand intuitively.

It shows that if an AF has some specific topology, we can judge its unstability and repairability in a simple and quick manner by virtue of topological features without regarding other semantics.

We think the range covered by the presented condition for unstability is enough wide, but there still remains room. In the future, we will consider the conditions for the stability of an AF or try to identify other conditions for unstability. We will also investigate more general conditions for 1-repair that makes the labels of some connectors *in*, and the repair of an AF including even-loops and one including arguments without attacks.

## References

1. Baroni, P., Caminada, M., Giacomin, M.: An introduction to argumentation semantics. Knowl. Eng. Rev. **26**(4), 365–410 (2011)
2. Baroni, P., Gabbay, D., Giacomin, M. (eds.): Handbook of Formal Argumentation. College Publications (2018)
3. Baroni, P., Giacomin, M., Guida, G.: Scc-recursiveness: A general schema for argumentation semantics. Artificial Intelligence **168**(1-2), 162–210 (2014)
4. Baroni, P., Giacomin, M., Liao, B.: On topology-related properties of abstract argumentation semantics. a correction and extension to dynamics of argumentation systems: A division-based method. Artificial Intelligence **212**, 104–115 (2014)
5. Baumann, R., Brewka, G.: Extension removal in abstract argumentation - an axiomatic approach. In: AAAI 2019. pp. 2670–2677 (2019)
6. Baumann, R., Brewka, G.: Expanding argumentation frameworks: Enforcing and monotonicity results. In: COMMA 2010. pp. 75–86 (2010)
7. Baumann, R., Brewka, G.: What does it take to enforce an argumeny? - minimal change in absrtact argumentatinon. In: ECAI 2012. pp. 127–132 (2012)
8. Baumann, R., Doutre, S., Mailly, J.G., Wallner, J.P.: Enforcement in formal argumentation. J. of Applied Logics **8**(6), 1623–1678 (2021)
9. Baumann, R., Gabbay, D.M., Rodrigues, O.: Forgetting an argument. In: AAAI 2020. pp. 2750–2757 (2020)
10. Baumann, R., Ulbricht, M.: If nothing is accepted - repairing argumentation frameworks. In: KR 2018. pp. 108–117 (2018)
11. Baumann, R., Ulbricht, M.: On cycles, attackers and supporters - a contribution to the investigation of dynamics in abstract argumentation. In: IJCAI 2021. pp. 1780–1786 (2021)
12. Bench-Capon, T., Dunne, P.E.: Argumentation in artificial intelligence. Artificial Intelligence **171**, 10–15 (2007)
13. Boella, G., Kaci, S., van der Torre, L.W.N.: Dynamics in argumentation with single extensions: Abstraction principles and the grounded extension. In: ECSQARU 2009. pp. 107–118 (2009)
14. Boella, G., Kaci, S., van der Torre, L.W.N.: Dynamics in argumentation with single extensions: attack refinement and the grounded extension. In: AAMAS 2009. pp. 1213–1214 (2009)
15. Cayrol, C., de Saint-Cyr, F.D., Lagasquie-Schiex, M.C.: Revision of an argumentation system. In: KR 2008. pp. 124–134 (2008)
16. Cayrol, C., de Saint-Cyr, F.D., Lagasquie-Schiex, M.C.: Change in abstract argumentation frameworks: Adding an argument. J. Artif. Intell. Res. **38**, 49–84 (2010)
17. Coste-Marquis, S., Konieczny, S., Mailly, J.G., Marquis, P.: Extension enforcement in abstract argumentation as an optimization problem. In: IJCAI 2015. pp. 2876–2882 (2015)

18. Doutre, S., Mailly, J.G.: Constraints and changes: A survey of abstract argumentation dynamics. Argument and Computation **9**(3), 223–248 (2018)
19. Dung, P.M.: On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games. Artificial Intelligence **77**, 321–357 (1995)
20. Liao, B., Jin, L., Koons, R.C.: Dynamics of argumentation systems: A division-based method. Artificial Intelligence **175**(11), 1790–1814 (2011)
21. Rahwan, I., Simari, G.R. (eds.): Argumentation in Artificial Intelligence. Springer (2009)
22. Saribatur, Z.G., Wallner, J.P.: Existential abstraction on argumentation frameworks via clustering. In: KR 2021. pp. 549–559 (2021)
23. Schulz, C., Toni, F.: On the responsibility for undecisiveness in preferred and stable labellings in abstract argumentation. Artificial Intelligence **262**, 301–335 (2018)
24. Takahashi, K.: Odd or even: Handling n-lemmas in a dynamic argumentation framework. In: SAFA 2022. pp. 5–18 (2022)
25. Takahashi, K., Okubo, T.: How can you resolve a trilemma? - a topological approach -. In: CLAR 2021. pp. 397–416 (2021)
26. Villata, S.: Explainable, Trustable and Emphatic Artificial Intelligence from Formal Argumentation Theory to Argumentation for Humans. habilitation, Université Côte D'Azur, Habilitation Thesis (2018)
27. Šefránek, J.: Updates of argumentation frameworks. In: NMR 2012 (2012)
28. Wallner, J.P., Niskanen, A., Järvisalo, M.: Complexity results and algorithms for extension enforcement in abstract argumentation. J. Artif. Intell. Res. **60**, 1–40 (2017)