Hybrid Reasoning on a Bipolar Argumentation Framework

Tatsuki Kawasaki, Sosuke Moriguchi, and Kazuko Takahashi

Kwansei Gakuin University, 2-1 Gakuen, Sanda, Hyogo, 669-1337, Japan
dxk96093@kwansei.ac.jp, chiguri@acm.org, ktaka@kwansei.ac.jp

Abstract. We develop a method of reasoning using an incrementally constructed bipolar argumentation framework (BAF) aiming to apply computational argumentation to legal reasoning. A BAF that explains the judgment of a certain case is constructed based on the user's knowledge and recognition. More specifically, a set of effective laws are derived as the conclusions from evidential facts recognized by the user, in a bottom-up manner; conversely, the evidences required to derive a new conclusion are identified if certain conditions are added, in a top-down manner. The BAF is incrementally constructed by repeated exercise of this bidirectional reasoning. The method provides support for those who are not familiar with the law, so that they can understand the judgment process and identify strategies that might allow them to win their case.

Keywords: Argumentation, Bidirectional reasoning, Legal reasoning.

1 Introduction

An argumentation framework (AF) is a powerful tool in the context of inconsistent knowledge [15, 21]. There are several possible application areas of AFs, including law [4, 20]. To date, research on applications has focused principally on AF updating to yield an acceptable set of facts when a new argument is presented, and strategies to win the argumentation when all of the dialog paths are known. However, in real legal cases, an AF representing a law in its entirety is usually incompletely grasped at the initial stage. Thus, it is more realistic to construct the AF incrementally; recognized facts are added in combination with AF reasoning.

For example, consider a case in which a person leased her house to another person, and the lessee then sub-leased a room to his sister; the lessor now wants to cancel the contract. (This is a simplified version of the case discussed in Satoh et al. [23].) The lessor decides to prosecute the lessee. The lessor knows that there was a lease, that they handed over the house to the lessee, and that the room was handed over by the lessee to the sublessee. However, if the lessor is not familiar with the law, she does not know what law might be applicable to her circumstances or what additional facts should be proven to make it effective. In addition, laws commonly include exceptions; that is, a law is effective if certain conditions are satisfied provided there is no exception.

For example, if there is no abuse of confidence, then the law of cancellation is not effective. Therefore, the lessor should check that there is "no abuse of confidence," as well as regarding facts that prove what must be proven. In addition, other facts may be needed to prove that there has been no abuse of confidence. Also, the presence of an exception may render another law effective. For those lacking a legal background, it can be difficult to grasp the entire structure of a particular law, which may be extensive and complicated. Thus, s/he often consults with, or even fully delegates the problem-solving process to, a lawyer. However, if the argumentation structure of the law was clear, s/he would be more likely to adequately understand the judgment process, obviating the need for a lawyer.

In this paper, we develop a bidirectional reasoning method using a bipolar argumentation framework (BAF) [2] that is applicable to legal reasoning. In a BAF, a general rule is represented as a support relation, and an exception as an attack relation. The facts of a case become arguments that are not attacked or supported by other arguments.

We explore the BAF in both a bottom-up and top-down manner, search for effective laws based on proven facts, and identify the facts required for application to other laws.

Beginning with the user-recognized facts of a specific case, laws that may be effective are searched for using a bottom-up process. Next, new conclusions are considered if specific conditions are satisfied. If such conclusions exist, the required facts are then identified in a top-down manner, so that the conditions are satisfied. If the existence of such facts can be proven, the facts are added as evidence, and the next round then begins. The procedure terminates if the user is satisfied with the conclusions obtained, or if no new conclusions are derived. By repeating this process, a user can simulate and scrutinize the judgment process to identify a strategy that may allow them to win the case.

This paper is organized as follows. In Section 2, we present the BAF, and the semantics thereof. In Section 3, we describe how the law is interpreted and represented using a BAF. In Section 4, we show the reasoning process of a BAF. In Section 5, we discuss related works. Finally, in Section 6, we present our conclusions and describe our planned future work.

2 Bipolar Argumentation Framework

A BAF is an extension of an AF in which the two relations of attack and support are defined over a set of arguments [2]. We define a support relation between a power set of arguments and a set of arguments; this differs from the common support relation of a BAF, so that it corresponds to a legal structure.

Definition 1 (bipolar argumentation framework). *A BAF is defined as a triple* $\langle AR, ATT, SUP \rangle$*, where* AR *is a finite set of arguments,* $ATT \subseteq AR \times AR$ *and* $SUP \subseteq (2^{AR} \setminus \{\emptyset\}) \times AR$*. If* $(B, A) \in ATT$*, then* B *attacks* A*; if* $(\mathbf{A}, A) \in SUP$*, then* $\mathbf{A}$ *supports* A*.*

A BAF can be regarded as a directed graph where the nodes and edges correspond to the arguments and the relations, respectively. Below, we represent a BAF graphically; a simple solid arrow indicates a support relation, and a straight arrow with a cutting edge indicates an attack relation. The dashed rectangle shows a set of arguments supporting a certain argument; it is sometimes omitted if the supporting set is a singleton.

Example 1. Figure 1 is a graphical representation of a BAF $\langle\{a, b, c, d, e\},\ \{(b, a), (e, d)\},\ \{(\{c, d\}, a)\}\rangle$.

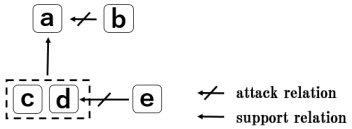


Fig. 1. Example of BAF.

Definition 2 (leaf). *An argument that is neither attacked nor supported by any other argument in a BAF is said to be* a leaf of the BAF.

For a BAF $\langle AR, ATT, SUP\rangle$, let $\rightarrow$ be a binary relation over AR as follows:

$$\rightarrow = ATT \cup \{(A, B) | \exists \mathbf{A} \subseteq AR, A \in \mathbf{A} \wedge (\mathbf{A}, B) \in SUP\}.$$

Definition 3 (acyclic). *A BAF $\langle AR, ATT, SUP\rangle$ is said to be* acyclic *if there is no $A \in AR$ such that $(A, A) \in \rightarrow^+$, where $\rightarrow^+$ is a transitive closure of $\rightarrow$.*

We define semantics for the BAF based on labeling [9]. Usually, labeling is a function from a set of arguments to $\{in, out, undec\}$, but *undec* is unnecessary here because we consider only acyclic BAFs. An argument labeled *in* is considered an acceptable argument.

Definition 4 (labeling). *For a BAF $\langle AR, ATT, SUP\rangle$, a* labeling $\mathcal{L}$ *is a function from AR to $\{in, out\}$.*

Labeling of a set of arguments proceeds as follows: $\mathcal{L}(\mathbf{A}) = in$ if $\mathcal{L}(A) = in$ for all $A \in \mathbf{A}$; and $\mathcal{L}(\mathbf{A}) = out$ otherwise.

Definition 5 (complete labeling). *For a BAF $baf = \langle AR, ATT, SUP\rangle$, labeling $\mathcal{L}$ is* complete *iff the following conditions are satisfied: for any argument $A \in AR$, (i) $\mathcal{L}(A) = in$ if A is a leaf* or $(\forall B \in AR; (B, A) \in ATT \Rightarrow \mathcal{L}(B) = out) \wedge (\exists \mathbf{A} \in 2^{AR}; (\mathbf{A}, A) \in SUP \wedge \mathcal{L}(\mathbf{A}) = in)$*, (ii) $\mathcal{L}(A) = out$ otherwise.*

If an argument is both attacked and supported, the attack is taken to be stronger than the support. For any acyclic BAF, there is exactly one complete labeling.

3 Description of Legal Knowledge in a BAF

In this paper, we consider an application of the Japanese civil code.

We assume that the BAFs are acyclic and that each law features both general rules and exceptions. A law is effective if the conditions of the general rule are satisfied unless an exception holds. We construct a BAF in which each condition in a rule is represented by an argument; the general rules can be represented by support relations, and the exceptions by attack relations. Therefore, our interpretations of attack and support relations differ from those used in the other BAFs. First, a support relation is defined as a binary relation of a power set and a set of arguments, since if one of the conditions is not met, the law is ineffective. Second, an argument lacking support is labeled *out*, even if it is attacked by an argument labeled *out*, since a law is not defined only by its exceptions and any argument other than a leaf should have an argument that supports it. The correspondence between the "acceptance" criterion of our BAF and that of a logic program is shown in [17].

We assume that the entire set of laws can be represented by a BAF termed *a universal BAF*, denoted as follows:

$$ubaf = \langle UAR, UATT, USUP \rangle.$$

It is almost impossible for a person who is not an expert to understand all of the laws. Therefore, we construct a specific BAF for each incident; relevant evidential facts are disclosed, and applicable laws identified using the universal BAF.

Definition 6 (existence/absence argument). *For an argument A, an argument showing the existence of an evidential fact for A is termed* an existence argument *and is denoted by* $\mathrm{ex}(A)$*; and an argument showing the absence of an evidential fact for A is termed* an absence argument *and is denoted by* $\mathrm{ab}(A)$*. These arguments are abbreviated as ex/ab arguments, respectively.*

Definition 7 (consistent ex/ab arguments set). *For a set of ex/ab arguments S, if there does not exist an argument A that satisfies both $ex(A) \in S$ and $ab(A) \in S$, then S is said to be* consistent*.*

Example 2. Figure 2 shows a BAF for the house lease case shown in Section 1, together with the relevant ex/ab arguments.

In this Figure, $\mathrm{ex}(a1)$, $\mathrm{ex}(a2)$, and $\mathrm{ex}(a4)$ are existence arguments for agreement_of_lease_contract, handover_to_lessee, and handover_to_ sublessee, respectively; $\mathrm{ab}(b1)$ is an absence argument for fact_of_non_abuse_of_confidence; and no evidence is currently shown for the other leaves.

4 Reasoning Using the BAF

4.1 Outline

We employ a running example throughout this section.

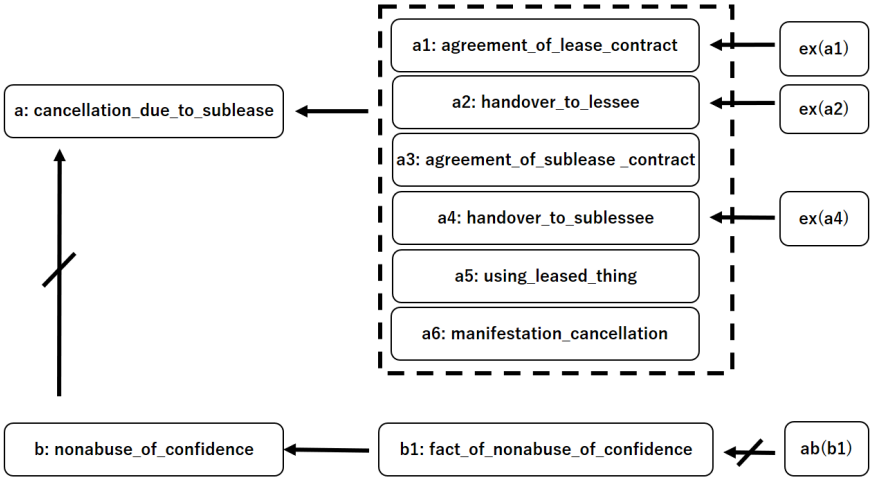


Fig. 2. Example of a BAF for a house-lease case.

Example 3. We assume the existence of the universal BAF $ubaf = \langle UAR, UATT, USUP \rangle$ shown in Figure 3.

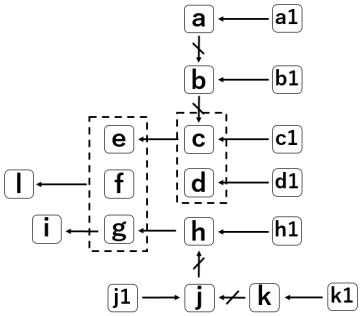


Fig. 3. Example of a universal BAF $ubaf$.

Let Ex be a set of ex/ab arguments that is currently recognized by a user. For either ex(A) or ab(A) of Ex, $A \in UAR$, and A is a leaf in $ubaf$.

Initially, a user recognizes a set of facts related to a certain incident. The reasoning proceeds by repeating two methods in turn. The first is used is to derive conclusions from the facts in a bottom-up manner, and the other is employed to find the evidence needed to draw a new conclusion if certain other conditions are met, this exercise proceeds in a top-down manner.

4.2 Bottom-up reasoning

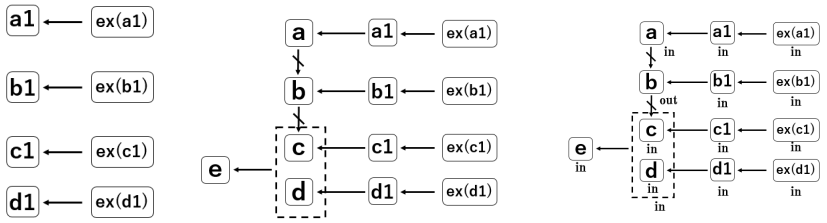
In bottom-up reasoning, arguments are derived by following the support relations from an ex/ab argument. The algorithm is shown in Algorithm 1.

Algorithm 1 BUP: find conclusions

Let Ex be a set of ex/ab arguments and $AR = \{A|\mathrm{ex}(A) \in Ex\} \cup \{A|\mathrm{ab}(A) \in Ex\}$.
Find a pair of a set of arguments $\mathbf{A} \subseteq AR$ and an argument $A \in UAR \setminus AR$ such that $(\mathbf{A}, A) \in USUP$.
while there exists such a pair $(\mathbf{A}, A)$ **do**
  Set $AR = AR \cup \{A\}$.
end while
Set $SUP = USUP \cap (2^{AR} \times AR) \cup \{(\{\mathrm{ex}(A)\}, A)|\mathrm{ex}(A) \in Ex\}$.
Set $ATT = UATT \cap (AR \times AR) \cup \{(\mathrm{ab}(A), A)|\mathrm{ab}(A) \in Ex\}$. Set $AR = AR \cup Ex$.
Apply the complete labeling $\mathcal{L}$ to $baf = \langle AR, ATT, SUP\rangle$.
$Concl(Ex) = \{A \mid \mathcal{L}(A) = in \wedge \neg\exists(\mathbf{A}, B) \in SUP; A \in \mathbf{A} \subseteq AR\}$.
return $Concl(Ex)$.

The resulting set of conclusions is the set of arguments that are acceptable, and no more conclusions can be drawn from the currently known facts.

Example 4. (Cont'd) Let Ex be $\{\mathrm{ex}(a1), \mathrm{ex}(b1), \mathrm{ex}(c1), \mathrm{ex}(d1)\}$, and *ubaf* be a BAF in Figure 3. Then, the BAF can be constructed using the process shown in Figure 4(a) and (b); finally, baf_1 is obtained, and $Concl(Ex) = \{a, e\}$ is derived as the set of conclusions. The complete labeling of the BAF baf_1 is shown in Figure 4(c).



(a) (b) (c)

Fig. 4. The bottom-up reasoning used to construct baf_1.

4.3 Top-down reasoning

On the other hand, we can seek additional facts that must be proven if a new conclusion is to be derived. Here, we search for a new conclusion and a set of supports, and identify the facts required to derive the arguments of the set.

Definition 8 (differential support pair, differential supporting set of arguments, differentially supported argument). *For a BAF* $baf = \langle AR, ATT, SUP \rangle$*, if* $(\mathbf{A} \cap AR) \neq \emptyset \wedge (\mathbf{A} \cap AR) \neq \mathbf{A} \wedge (\mathbf{A}, A) \in USUP$*, then* $(\mathbf{A} \setminus AR, A)$ *is said to be* a differential support pair on baf. In addition, $\mathbf{A} \setminus AR$ and A are said to be *a differential supporting set of arguments on* baf, and *a differentially supported argument on* baf, respectively.

Intuitively, differential support pair means that A cannot be derived using the current BAF due to the lack of required conditions, but it can be derived if all of the arguments in $\mathbf{A} \setminus AR$ are accepted. In general, there may exist several differential support pairs on any BAF.

Example 5. (Cont'd) For baf_1, we find differential support pair $(\{f, g\}, l)$, because $\{e, f, g\} \cap AR = \{e\} \neq \emptyset$ and $(\{e, f, g\}, l) \in USUP$ (Figure 5).

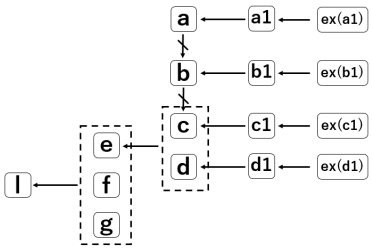


Fig. 5. A differential support pair on baf_1: $(\{f, g\}, l)$.

For a BAF $baf = \langle AR, ATT, SUP \rangle$ and an argument $A \in AR$, we detect a set of facts that satisfies $\mathcal{L}(A) = in$. For an argument A, we check the conditions for labeling of the arguments that attack A and the sets of arguments that support A. This is achieved by repeatedly applying the following two algorithms: $PC(A)$ and $NC(A)$, which are shown in Algorithm 2 and Algorithm 3, respectively. Note that there is no argument that both lacks support and is attacked.

Then, discovery of the required facts proceeds using the algorithm shown in Algorithm 4.

As a result, a set of ex/ab arguments is generated. An existence argument $\mathrm{ex}(A)$ shows that the fact is required if $\mathcal{L}(\mathbf{A}) = in$ is to hold, whereas an absence argument $\mathrm{ab}(A)$ shows that the evidence is an obstacle to prove $\mathcal{L}(\mathbf{A}) = in$.

Example 6. (Cont'd) For a differential supporting set of arguments $\{f, g\}$, we find $Sol(\{f, g\}) = PC(f) \cup PC(g)$.
(i) $PC(f) = \{\mathrm{ex}(f)\}$.
(ii) $PC(g) = PC(h) = PC(h1) \cup NC(j)$ (Figure 6). As for $PC(h1)$, we obtain $\{\mathrm{ex}(h1)\}$. As for $NC(j)$, we have two alternatives: $NC(j1)$ and $PC(k)$ (Figure 7).

Algorithm 2 $PC(A)$: find required arguments for $\mathcal{L}(A) = in$.

Let A be an argument in UAR.
if A is a leaf of *ubaf* **then**
  $Sol(A) = \{\mathrm{ex}(A)\}$.
else
  Choose an arbitrary set of arguments $\mathbf{A}$ that satisfies $(\mathbf{A}, A) \in USUP$.
  $Sol(A) = \bigcup_{(B,A)\in UATT} NC(B) \cup \bigcup_{A_i\in\mathbf{A}} PC(A_i)$.
end if
return $Sol(A)$.

Algorithm 3 $NC(A)$: find required arguments for $\mathcal{L}(A) = out$.

Let A be an argument in UAR.
if A is a leaf of *ubaf* **then**
  $Sol(A) = \{\mathrm{ab}(A)\}$.
else
  Choose an arbitrary argument B that satisfies $(B, A) \in UATT$.
  Let $\mathbf{A}_1, \ldots, \mathbf{A}_n$ be all sets of arguments such that $(\mathbf{A}_i, A) \in USUP (i = 1, \ldots, n)$.
  Choose an arbitrary argument $A_i \in \mathbf{A}_i$ $(i = 1, \ldots, n)$.
  Either $Sol(A) = PC(B)$
    or $Sol(A) = \bigcup_{i=1,\ldots,n} NC(A_i)$.
end if
return $Sol(A)$.

Assume that we choose the condition $NC(j1)$. Then, we find $\{\mathrm{ab}(j1)\}$ as $Sol(j1)$ (Figure 8). Finally, we obtain a set of required facts $\{\mathrm{ex}(f), \mathrm{ex}(h1), \mathrm{ab}(j1)\}$ (Figure 9).

4.4 Hybrid reasoning

The algorithm used for hybrid reasoning is Algorithm 5.

As a result, the required facts are identified, and conclusions are derived from these facts.

Example 7. (Cont'd) For a set of required facts $\{\mathrm{ex}(f), \mathrm{ex}(h1), \mathrm{ab}(j1)\}$, assume that a user has confirmed the existence of f and $h1$, and the absence of $j1$. Then, we construct a new BAF baf_2 in a bottom-up manner from this set. Part of the labeling of baf_2 is shown in Figure 10. Finally, we obtain a new conclusion set $Concl = \{a, i, l\}$.

Algorithm 4 TDN: find required facts

Let $baf = \langle AR, ATT, SUP\rangle$ be a BAF and $\mathbf{A}$ a differential supporting set of arguments on baf.
$Sol(\mathbf{A}) = \bigcup_{A\in\mathbf{A}} PC(A)$.
return $Sol(\mathbf{A})$.

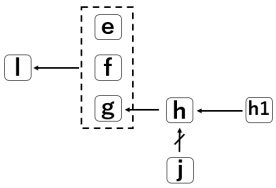


Fig. 6. Top-down reasoning: Both $NC(j)$ and $PC(h1)$ are required.

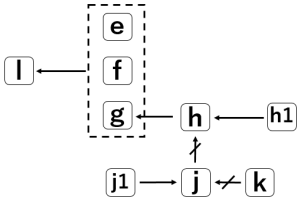


Fig. 7. Top-down reasoning: Either $NC(j1)$ or $PC(k)$ is required.

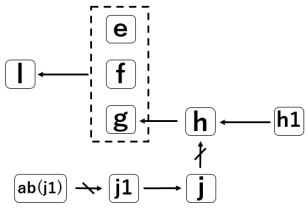


Fig. 8. Top-down reasoning: The situation when choosing $NC(j1)$.

The hybrid algorithm is nondeterministic at several steps and there are multiple possible solutions.

Example 8. (Cont'd) Assume that we choose the condition $PC(k)$ in Figure 7. Then, we find $\{\mathrm{ex}(k1)\}$ as $Sol(k1)$, and the set of required facts is $\{\mathrm{ex}(f), \mathrm{ex}(h1), \mathrm{ex}(k1)\}$. In this case, we construct the different BAF baf'_2 shown in Figure 11 after a second round of bottom-up reasoning. Strictly speaking, an argument j and the attack relations (k, j) and (j, h) do not appear in baf'_2 because a new argument is created by tracing only a support relation in BUP. However, it is reasonable to show the attack relation traced in the TDN, considering that the BAF is constructed based on the user's current knowledge. Note that these attacks do not affect the label $\mathcal{L}(h) = in$.

4.5 Correctness

We now prove the validity of hybrid reasoning.

In the proof, we use the height of an argument in *ubaf*, as defined in [17].

Definition 9. *For the acyclic universal BAF ubaf, the height of an argument A is defined as follows:*

- *If A is a leaf, then the height of A is* 0.
- *If there are some arguments B such that* $(B, A) \in \rightarrow$*, then the height of A is* $h + 1$*, where h is the maximum height of this B.*

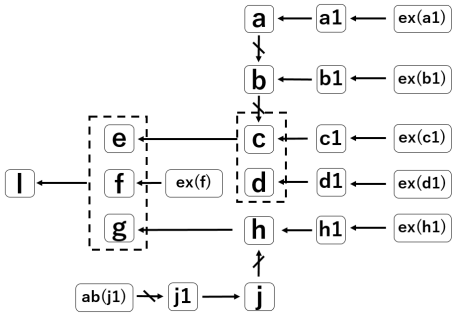


Fig. 9. Top-down reasoning: $Ex = \{\mathrm{ex}(f), \mathrm{ex}(h1), \mathrm{ab}(j1)\}$.

Algorithm 5 HR: hybrid reasoning

Let Ex be a set of ex/ab arguments in an initial state.
For Ex, obtain $Concl(Ex)$ and a new *baf* by BUP.
while a user does not attain a goal that satisfies him/her, and TDN returns a consistent solution with Ex **do**
  For a *baf* and an arbitrary $\mathbf{A}$ on *baf*, obtain $Sol(\mathbf{A})$ by TDN.
  for each $\mathrm{ex}(A)$ or $\mathrm{ab}(A)$ in $Sol(\mathbf{A})$ **do**
    Ask the user to confirm that existence or absence.
    if there exists a fact for A **then**
      Set $Ex = Ex \cup \{\mathrm{ex}(A)\}$.
    else
      Set $Ex = Ex \cup \{\mathrm{ab}(A)\}$.
    end if
    Get $Concl(Ex)$ and a new *baf* by BUP.
  end for
end while

It is easy to show that the heights of arguments are definable when *ubaf* is acyclic.

Here, we prove two specifications, one for a BUP, and the other for a TDN. For a BUP, the built BAF includes arguments pertaining to the evidential facts that the user recognizes. Notably, the acceptability of such arguments is the same as that of the universal BAF.

Theorem 1. *Assume that ubaf is acyclic. Let baf be built by BUP from Ex. When UEx is defined as* $\{\mathrm{ex}(A)|\mathrm{ex}(A) \in Ex\} \cup \{\mathrm{ab}(A)|A$ *is a leaf of ubaf* $\wedge$ $\mathrm{ex}(A) \notin Ex\}$*, and* $\mathcal{L}_U$ *is a complete labeling for* $\langle UAR \cup UEx, UATT \cup \{(\mathrm{ab}(A), A)|\mathrm{ab}(A) \in UEx\}, USUP \cup \{(\{\mathrm{ex}(A)\}, A)|\mathrm{ex}(A) \in UEx\}\rangle$*, for any argument* $A \in UAR$*,* $A \in AR \wedge \mathcal{L}(A) = \mathcal{L}_U(A)$*, or* $A \notin AR \wedge \mathcal{L}_U(A) = out$*.*

Proof. We prove this by induction on the height of A. When A is a leaf, if $\mathrm{ex}(A) \in Ex$ (i.e., $\mathrm{ex}(A) \in UEx$), then $A \in AR$ and $\mathcal{L}(A) = \mathcal{L}_U(A) = in$. If $\mathrm{ex}(A) \notin Ex$ (i.e., $\mathrm{ab}(A) \in UEx$), then $\mathcal{L}_U(A) = out$. In this case, if $\mathrm{ab}(A) \in Ex$

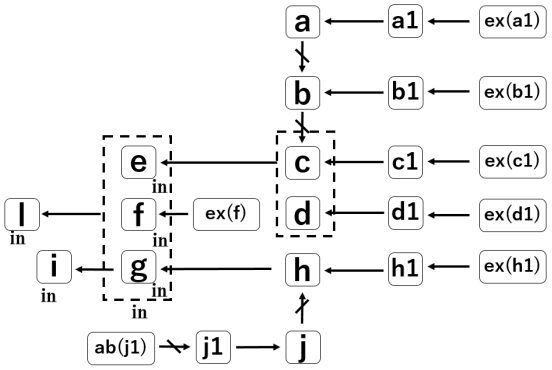


Fig. 10. The BAF obtained after the second round of bottom-up reasoning: baf_2.

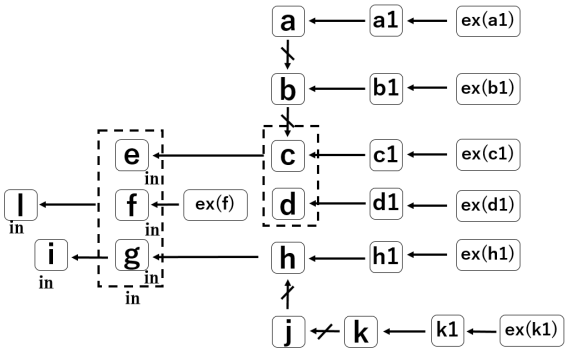


Fig. 11. The BAF obtained after the second round of bottom-up reasoning: baf'_2.

then $A \in AR$ but $\mathcal{L}(A) = out$, and, otherwise $A \notin AR$. Both cases satisfy the proposition.

Assume that A is not a leaf. If $\mathcal{L}_U(A) = in$, then there are some supports $(\mathbf{A}, A) \in USUP$ such that $\mathcal{L}_U(\mathbf{A}) = in$, and any attacks $(B, A) \in UATT$, $\mathcal{L}_U(B) = out$. From the induction hypothesis, for any $C \in \mathbf{A}$, $C \in AR$, and $\mathcal{L}(C) = in$; and for any attackers B of A, $\mathcal{L}(B) = out$ or $B \notin AR$. The definition of BUP immediately shows that $A \in AR$, and therefore $\mathcal{L}(A) = in = \mathcal{L}_U(A)$.

Assume that $\mathcal{L}_U(A) = out$. If $A \notin AR$, the proposition is satisfied. Otherwise, $A \in AR$, and from the definition of BUP, there are some supports $(\mathbf{A}, A)$ such that $\mathbf{A} \subseteq AR$, so A is not a leaf of baf. From $\mathcal{L}_U(A) = out$, there are some attacks $(B, A) \in UATT$ such that $\mathcal{L}_U(B) = in$, or for any supports $(\mathbf{A}, A) \in USUP$, $\mathcal{L}_U(A) = out$ (i.e., there exists $C \in \mathbf{A}$ such that $\mathcal{L}_U(C) = out$). From the induction hypothesis, there are some attacks $(B, A) \in UATT$ such that $\mathcal{L}(B) = in$, or for any supports $(\mathbf{A}, A) \in USUP$, there exists $C \in \mathbf{A}$ such that $C \notin AR$ or

$\mathcal{L}(C) = out$. For the former case, $(B, A) \in ATT$, and therefore $\mathcal{L}(A) = out$. For the latter case, for any $(\mathbf{A}, A) \in SUP$, $\mathcal{L}(\mathbf{A}) = out$, and therefore $\mathcal{L}(A) = out$.

From the above, $A \in AR \wedge \mathcal{L}(A) = \mathcal{L}_U(A)$, or $A \notin AR \wedge \mathcal{L}_U(A) = out$. □

For a TDN, the facts found by $PC(A)$ make the argument A acceptable.

Theorem 2. *Assume that ubaf is acyclic and that A is an argument in UAR. If $Ex \cup PC(A)$ is consistent and baf is built by BUP from $Ex \cup PC(A)$, then $A \in AR$, and the complete labeling $\mathcal{L}$ satisfies $\mathcal{L}(A) = in$. If $Ex \cup NC(A)$ is consistent and baf is built by BUP from $Ex \cup NC(A)$, then $A \notin AR$, or $A \in AR$ and the complete labeling $\mathcal{L}$ satisfies $\mathcal{L}(A) = out$.*

Proof. We prove this by induction on the height of A. For the former case, assume that $Ex \cup PC(A)$ is consistent. When A is a leaf (thus of height 0), $PC(A) = \{\mathrm{ex}(A)\}$ (i.e., *baf* includes A and $\mathrm{ex}(A)$), and therefore, $\mathcal{L}(A) = in$. Otherwise, for some $\mathbf{A}$ satisfying $(\mathbf{A}, A) \in USUP$, $PC(A) = \bigcup_{(B,A)\in UATT} NC(B) \cup \bigcup_{C\in\mathbf{A}} PC(C)$. For each B such that $(B, A) \in ATT$, $NC(B) \subseteq PC(A)$, and $Ex \cup NC(B)$ is thus consistent. As the height of B is less than that of A, from the induction hypothesis, $B \notin AR$ or $B \in AR$ but $\mathcal{L}(B) = out$. In a similar fashion, for each $C \in \mathbf{A}$, $C \in AR$ and $\mathcal{L}(C) = in$, and therefore $\mathcal{L}(\mathbf{A}) = in$. From the definitions of BUP and complete labeling, $A \in AR$ and $\mathcal{L}(A) = in$.

The proof for the case of $NC(A)$ is the same. □

5 Related Works

Support relations play important roles in our approach. Such relations can be interpreted in several ways [12]. Cayrol et al. defined several types of indirect attacks by combining attacks with supports, and defined several types of extensions in BAF [10]. Boella et al. revised the semantics by introducing different meta-arguments and meta-supports [6]. Noueioua et al. developed a BAF that considered a support relation to be a "necessity" relation [18]. Čyras et al. considered that several semantics of a BAF could be captured using assumption-based argumentation [13]. Brewka et al. developed an abstract dialectical framework (ADF) as a generalization of Dung's AF [7, 8]; a BAF was represented using an ADF. These works focus on acceptance of arguments. Here, we define a support relation and develop semantics that can represent a law.

Several authors have studied changes in AFs when arguments are added or deleted [14]. Cayrol et al. investigated changes in acceptable arguments when an argument was added to a current AF [11]. Baumann et al. developed a strategy for AF diagnosis and repair, and explored the computational complexity thereof [3]. Most research has focused on semantics, and changes in acceptable sets when arguments are added/deleted. The computational complexity associated with AF updating via argument addition/deletion is a significant issue [1]. Here, we propose the reasoning based on an incrementally constructed BAF, potentially broadening the applications of such frameworks. Complexity is not of concern; we do not need to consider all possibilities since solutions can be derived from

a given universal BAF. However, it is possible to use efficient computational methods when executing our algorithm.

Our reasoning mechanism may be considered a form of hypothetical reasoning, or an abduction, which is a method used to search for the set of facts necessary to derive an observed conclusion [19]. In assumption-based argumentation, abduction is used to explain a conclusion supported by an argument [5]. Combinations of abduction and argumentation have been discussed in several works. Kakas et al. developed a method to determine the conditions that support arguments [16]. Sakama studied an abduction in argumentation framework [22] and proposed a method to search for the conditions explaining the justification state. This may include removal of an argument if it is not justified. Also, a computational method was developed by transforming an AF into a logic program. In our approach, we do not remove arguments; instead, we add absence arguments, which is equivalent to argument removal. It is reasonable to confirm the existence or absence of evidential facts when aiming to establish whether a certain law applies. The difference between the cited works and our method is that, in the previous works, observations are given and the facts that can explain those arguments are searched. In our case, potential conclusions justified by the observed facts are not specified; instead, bidirectional reasoning is performed repeatedly to assemble a knowledge set in an incremental manner. In addition, the purpose of our research is to support simulations. A minimal set of facts does not necessarily yield the best solution, unlike the cases of conventional hypothetical reasoning and common abduction.

6 Conclusion

In this paper, we developed a hybrid method featuring both bottom-up and top-down reasoning using an incrementally constructed BAF. The method can be applied to find a relevant law based on proven facts, and suggests facts that might make another law applicable. The proposed method can support those who are not familiar with a law through a simulation process, allowing a better understanding of the law to be achieved, in addition to identifying potential strategies for winning the case.

We are currently exploring reasoning processes that use three-valued representation, of which *undecided* is one possible representation. In future, we plan to implement visualization of our method.

Acknowledgment. This work was supported by JSPS KAKENHI Grant Number JP17H06103.

References

1. Alfano, G., Greco, S. and Parisi, F.: A meta-argumentation approach for the efficient computation of stable and preferred extensions in dynamic bipolar argumentation frameworks. *Intelligenza Artificiale*, **12**(2), 193–211 (2018).

2. Amgoud, L., Cayrol, C., Lagasquie-Schiex, M. C. and Livet, P.: On bipolarity in argumentation frameworks. *International Journal of Intelligent Systems*, **23**(10), 1062–1093 (2008).
3. Baumann, R. and Ulbricht, M.: If nothing is accepted - Repairing argumentation frameworks. In *Proc. of KR2010*, 108–117 (2018).
4. Bench-Capon, T., Prakken H. and Sartor, G.: Argumentation in legal reasoning. *Argumentation in Artificial Intelligence*, 363–382 (2009).
5. Bondarenko, A., Dung, P. M., Kowalski, R. and Toni, F.: An abstract, argumentation-theoretic approach to default reasoning. *Artificial Intelligence*, **93**, 63–101 (1997).
6. Boella, G., Gabbay, D. M., Torre, L. van der and Villata, S.: Support in abstract argumentation In *Proc. of COMMA2010*, 40–51 (2010).
7. Brewka, G. and Woltran, S.: Abstract dialectical frameworks. In *Proc. of KR2010*, 102-111 (2010).
8. Brewka, G., Ellmauthaler, S., Strass, H., Wallner, J.P. and Woltran, S.: Abstract dialectical frameworks revisited. In *Proc. of IJCAI2013*, 803–809 (2013).
9. Caminada, M.: On the issue of reinstatement in argumentation. In *Proc. of JELIA2006*, 111–123 (2006).
10. Cayrol, C. and Lagasquie-Schiex, M.: On the acceptability of arguments in bipolar argumentation frameworks. In *Proc. of ECSQARU2005*, 378–389 (2005).
11. Cayrol, C., de Saint-Cyr, F. D. and Lagasquie-Schiex, M.: Change in abstract argumentation frameworks: Adding an argument. *Journal of Artificial Intelligence Research*, **28**, 49–84 (2010).
12. Cohen, A., Gottifredi, S., Garcia, A. and Simari, G.: A survey of different approaches to support in argumentation systems. *The Knowledge Engineering Review*, **29**(5), 513–550 (2013).
13. Čyras, K., Schulz, C. and Toni, F.: Capturing bipolar argumentation in non-flat assumption-based argumentation. In *Proc. of PRIMA2017*, 386–402 (2017).
14. Doutre, S. and Jean-Guyb, M.: Constraints and changes: A survey of abstract argumentation dynamics. *Argument an Computation*, **9**(3), 223–248 (2018).
15. 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).
16. Kakas, A. C. and Moraitis, P. Argumentative agent deliberation, roles and context. *Electronic Notes in Theoretical Computer Science*, **70**, 39–53 (2002).
17. Kawasaki, T., Moriguchi, S. and Takahashi, K.: Transformation from PROLEG to a bipolar argumentation framework. In *Proc. of SAFA2018*, 36–47 (2018).
18. Nouioua, F. and Risch, V.: Argumentation framework with necessities. In *Proc. of SUM2011*, 163–176 (2011).
19. Poole, D.: Logical framework for default reasoning. *Artificial Intelligence*,**36**, 27–47 (1988).
20. Prakken, H. and Sartor, G: Law and logic: A review from an argumentation perspective. *Artificial Intelligence*, **36**, 214–245 (2015).
21. Rahwan, I. and Simari, G.(eds.): *Argumentation in Artificial Intelligence*, Springer (2009).
22. Sakama, C: Abduction in argumentation frameworks. *Journal of Applied Non-Classical Logics*, **28**, 218–239 (2018).
23. Satoh, K. et al.: PROLEG: An Implementation of the Presupposed Ultimate Fact Theory of Japanese Civil Code by PROLOG Technology. In *JSAI-isAI 2010: New Frontiers in Artificial Intelligence*, 153–164 (2010).