Argument Aggregation: Basic Axioms and Complexity Results

Paul E. DUNNE\textsuperscript{a} Pierre MARQUIS\textsuperscript{b} Michael WOOLDRIDGE\textsuperscript{c}
\textsuperscript{a} Department of Computer Science, The University of Liverpool, UK
\textsuperscript{b} CRIL-CNRS/Universite d’Artois, Lens, France
\textsuperscript{c} Department of Computer Science, University of Oxford, UK

Abstract. Argument aggregation is the problem of combining argumentation frameworks. An argument aggregation procedure takes as input an argument framework for each agent in a system, intuitively representing the beliefs of that agent with respect to a disputed domain of discourse; the output is an argumentation framework that represents the social position on the domain of discourse. There are clear analogies between argument aggregation and the well-known preference aggregation problem, which has been extensively studied in the social choice community. The first contribution of this paper is to apply some of the methodology developed in social choice theory to argument aggregation. After recalling the basic framework of Dung’s abstract argument systems, and introducing the argument aggregation problem, we motivate and formally define a collection of axioms that specific argument aggregation procedures might or might not satisfy. The second contribution of the paper is to consider the analysis of argument aggregation procedures with respect to these various axioms. We consider a natural representation for argument aggregation procedures, based on Boolean circuits. We then investigate the problem of verifying whether an argument aggregation procedure, presented in this way, does or does not satisfy a number of the axioms we introduced.

Keywords. Computational properties of argumentation; Argumentation Frameworks; aggregation; computational complexity

Introduction

One of the fundamental challenges in multi-agent systems research is that of coping with inconsistencies between the beliefs and/or preferences of agents [2]. Social choice theory studies the problem of making satisfactory social decisions in the presence of conflicting desires or preferences over the space of decisions [1]. The problem of combining preferences/desires in order to make a social choice is known as the preference aggregation problem. One key theme of research in social choice theory is to characterise axiomatically various desirable and undesirable properties that a preference aggregation procedure might satisfy. Given such axioms, one can then investigate the extent to which actual social choice procedures do or do not satisfy these axioms. Another fundamental line of research is to investigate the extent to which desirable axioms are mutually satisfiable or indeed mutually exclusive: Arrow’s theorem [3], one of the most famous results in social choice theory, is exactly such a result, and states that there can be no social choice procedure that satisfies some basic “reasonableness” axioms.
The first contribution of the present paper is to begin to apply some of the methodology of social choice theory described above – in particular, the axiomatic characterisation of desirable/undesirable properties – to the problem of argument aggregation. In an argument aggregation problem, we have a set of agents, where each agent is associated with an argumentation framework [6,13], which, intuitively, represents that agent’s beliefs. It is assumed that the beliefs of different agents may be in conflict. The goal is to aggregate these potentially different argumentation frameworks in order to derive an overall social argumentation framework. The problem can be seen as analogous to preference aggregation: instead of aggregating preference profiles, we are aggregating argumentation frameworks. This analogy suggests the first problem to which we address ourselves in the present paper, that is, what axioms should an argument aggregation procedure satisfy? To this end, we motivate and formally define a number of potential axioms for argument aggregation. As we will see, the presentation of these axioms is more complex than for preference aggregation because argumentation frameworks are interpreted with respect to a particular semantics. For Dung’s abstract argumentation systems, for example, possible semantics include preferred extensions, grounded extensions, stable sets, and so on [6]. For any given argumentation framework, these different semantics will, in general, yield different sets of accepted arguments. Thus, we cannot study argument aggregation axioms without reference to the possible semantics.

The second aim of the paper is to consider the problem of analysing argument aggregation procedures with respect to these axioms. That is, we consider the problem of checking whether specific argument aggregation procedures do or do not satisfy various of the axioms we introduce. In order to consider this problem, we need a concrete computational representation for argument aggregation procedures. We consider a natural representation, based on Boolean circuits. We then investigate the problem of verifying whether an argument aggregation procedure, presented in this way, does or does not satisfy a number of the axioms we introduced.

1. Basic Definitions

In this section, we present the main definitions that are used throughout the remainder of the paper. As our argument aggregation framework is ultimately built on the framework of Dung’s abstract argument systems [6], we will begin by recalling the main elements of this framework, and some basic facts about it.

1.1. Dung-style Abstract Argument Systems

Let \( \mathcal{X} = \{x_1, \ldots, x_m\} \) be a finite, non-empty set of atomic arguments. A Dung style argument system (hereafter simply “argument system”) over \( \mathcal{X} \) is defined by a binary attack relation \( A \subseteq \mathcal{X} \times \mathcal{X} \). The intended interpretation of the attack relation is as follows: if \( (x, x') \in A \), then if argument \( x \) is accepted as an outcome of argumentation, then \( x' \) should not be accepted. We usually write “\( x \) attacks \( x' \)” when \( (x, x') \in A \). We will use \( \varepsilon, \varepsilon', \ldots \) as variables ranging over attacks when we are not concerned with the source and target arguments; when we need to refer to these elements, we will write the attack in full, as in \( (x, x') \).

We use \( \Delta(\mathcal{X}) \) to denote the set of all argument systems over \( \mathcal{X} \). Where \( \mathcal{X} \) is unambiguous from context, we omit reference to it and write \( \Delta \). For \( n \in \mathbb{N} \) such that \( n > 0 \),
we write $\Delta^n$ to denote $\Delta \times \cdots \times \Delta$ with $\langle A_1, \ldots, A_n \rangle$ used for explicit reference to elements of such an n-tuple.

Where $H = \langle \mathcal{X}, A \rangle$ is an argument system and $S \subseteq \mathcal{X}$ a subset of arguments, then we say that $S$ is: conflict free if for no $\{x, x'\} \subseteq S$ is it the case that $(x, x') \in A$; defensive if for every $(x, x') \in A$ such that $x' \in S$, there exists some $x'' \in S$ (possibly $x' = x''$) such that $(x'', x) \in A$; admissible if it is both conflict free and defensive; a preferred extension if it is a maximal (w.r.t. $\subseteq$) admissible set; and a stable extension if it is an admissible set which attacks all arguments in $\mathcal{X} \setminus S$. We assume the standard definition of grounded extensions [6].

It will be useful to describe the general concepts below in terms of arbitrary semantics. Formally, we view a semantics as a function $\sigma : \Delta \rightarrow 2^\Delta$.

Given a semantics $\sigma$, let $E_\sigma(H)$ denote the $\sigma$-solutions of argument system $H \in \Delta$. We use $pe$ to denote preferred extensions, and $gnd$ to denote grounded extensions. Note that for $\sigma \in \{pe, gnd\}$, $\forall H \in \Delta, |E_\sigma(H)| \geq 1$, although $\exists H \in \Delta$ such that $E_\sigma(H) = \{\emptyset\}$. In the case of stable extensions ($\sigma = st$), however, $\exists H \in \Delta$ such that $E_{st}(H) = \emptyset$.

If $|E_\sigma(H)| = 1$ and $\emptyset \notin E_\sigma(H)$ then we say $H$ is $\sigma$-decisive. Let $\Delta_{D,\sigma}(X)$ denote the set of decisive argument systems over $X$ – again we drop reference to $X$ where this is clear. The requirement that $H$ is acyclic is sufficient to guarantee decisiveness w.r.t. $\sigma \in \{pe, gnd, st\}$ (although it is not a necessary condition).

We say that an argument system $H$ is $\sigma$-non-trivial (for cases where $E_\sigma$ is always non-empty), if $E_\sigma(H) \neq \{\emptyset\}$. Note that $\sigma$-decisiveness implies $\sigma$-non-triviality, although of course the two conditions are not equivalent. Let $\Delta_{NT,\sigma}(X)$ denote the set of $\sigma$-non-trivial argument systems over $X$ (we write $\Delta_{NT,\sigma}$, dropping reference to $X$, where there is no ambiguity).

We say an argument $x \in \mathcal{X}$ is credulously accepted for $\sigma$ w.r.t. $H$ if it is a member of at least one set in $E_\sigma(H)$ and we say $x$ is sceptically accepted for $\sigma$ if it is a member of every set in $E_\sigma(H)$. Let $ca_\sigma$ denote the credulously accepted for $\sigma$ arguments of $H$, and let $sa_\sigma(H)$ denote the sceptically accepted for $\sigma$ arguments of $H$:

$$
c_{a_\sigma}(H) = \bigcup_{S \in E_\sigma(H)} S \quad s_{a_\sigma}(H) = \bigcap_{S \in E_\sigma(H)} S
$$

1.2. Argument Aggregation

We can now define argument aggregation. Let $\mathcal{N} = \{1, \ldots, n\}$ be a finite, non-empty set of agents. Each agent $i \in \mathcal{N}$ is assumed to be associated with an argument system $H_i \in \Delta$, and we sometimes refer to $H_i$ as the position of agent $i$. Intuitively, $H_i$ represents $i$’s beliefs about the relationships between the arguments $\mathcal{X}$. Each preferred extension $S \in pe(H_i)$ then represents one “rational interpretation” of agent $i$’s beliefs – the fact that there is more than one preferred extension intuitively captures agent $i$’s uncertainty about the status of some arguments. Thus the arguments in $sa(H_i)$ represent arguments whose status the agent $i$ is certain about.

In the everyday world, it is possible that agents may have the same beliefs (i.e., have the same preferred extensions), but for different reasons (i.e., have different attack relations). More usually, however, they will have different attack relations and different preferred extensions. Put simply, the argument aggregation problem is the problem of
reconciling these different argument systems, to obtain a single social argument system, or social position, which we often denote by \( H^* \). The basic issue we address in the remainder of this paper is how to aggregate argument systems in a principled, rational, and democratic way.

Formally, an argument aggregation domain is a structure: \( \langle \mathcal{X}, \mathcal{N}, H_1, \ldots, H_n \rangle \) where \( \mathcal{X} = \{ x_1, \ldots, x_m \} \) is a set of arguments, \( \mathcal{N} = \{ 1, \ldots, n \} \) is a set of agents, and for all \( i \in \mathcal{N} \), \( H_i \in \Delta \) is the position of agent \( i \) – an argument system over \( \mathcal{X} \).

We model argument aggregation through argument aggregation functions. An argument aggregation function \( \gamma \) has the signature: \( \gamma : \Delta \to \Delta \) (and we use \( \hat{\mathcal{H}} \) for a typical tuple in \( \Delta^n \)). Let \( \Gamma(\mathcal{X}, \mathcal{N}) \) be the set of all argument aggregation functions over \( \mathcal{X}, \mathcal{N} \) – again, where context makes \( \mathcal{X}, \mathcal{N} \) clear, we write \( \Gamma \).

2. Axioms

As we noted earlier in this paper, a key concern in the social choice research domain is that of axiomatically classifying desirable and undesirable properties of social welfare functions, and then investigating the extent to which actual social welfare functions can and do satisfy these properties. One of the key aims of this paper is to identify the range of potential properties of argument aggregation functions. Roughly speaking, we can distinguish between these properties on the basis of whether they appeal to the ideals of either rationality or democracy. Rationality properties state that the outcome of aggregation should “make sense” from a rationality perspective, while democracy properties state that the outcome should reflect the beliefs of the participants. We now, briefly, expand on these.

2.1. Anonymity

Anonymity says that the argument aggregation process should not take into account the identities of the agents submitting positions when computing a social argument system. Let \( \Pi(s) \) be all permutations of the tuple \( s \). An argument aggregation function \( \gamma \) is anonymous (ANON) if it produces the same social argument system for all permutations of the same input.

\[ \forall \hat{\mathcal{H}} \in \Delta^n : \forall \hat{\mathcal{H}}' \in \Pi(\hat{\mathcal{H}}) : \gamma(\hat{\mathcal{H}}) = \gamma(\hat{\mathcal{H}}'). \]

2.2. Non-Triviality

An argument aggregation function is said to be non-trivial if it is guaranteed to produce a “meaningful” answer, i.e. it is guaranteed to produce as output a non-trivial social argument system. The argument aggregation function \( \gamma \) is \( \sigma \)-strongly non-trivial (\( \sigma \)-SNT) if:

\[ \forall \hat{\mathcal{H}} \in \Delta^n : \gamma(\hat{\mathcal{H}}) \in \Delta_{NT,\sigma} \]

This is a very strong property. For example, consider the case where every agent \( i \in \mathcal{N} \) presents a trivial argument system \( H_i \), i.e., such that \( E_\sigma(H_i) = \{ \emptyset \} \). In this case, there is no sensible social position. For this reason, we define an argument aggregation function
as \( \sigma \)-weakly non-trivial (\( \sigma \)-WNT) if, for every input in which every agent presents a non-trivial argument system, the argument aggregation function is guaranteed to produce a non-trivial output:

\[
\forall \hat{H} \in \Delta^n_{NT,\sigma} : \gamma(\hat{H}) \in \Delta_{NT,\sigma}.
\]

### 2.3. Decisiveness

Decisiveness is somewhat related to non-triviality. The basic idea is that an argument aggregation function is decisive if it is guaranteed to produce as output a decisive social argument system. The \( \sigma \)-strong decisiveness (\( \sigma \)-SD) property of \( \gamma \) is characterised as follows:

\[
\forall \hat{H} \in \Delta^n : \gamma(\hat{H}) \in \Delta_{D,\sigma}.
\]

By a similar rationale to that presented earlier, this may be too strong in general: hence, we consider \( \sigma \)-weak decisiveness (\( \sigma \)-WD):

\[
\forall \hat{H} \in \Delta^n_{D,\sigma} : \gamma(\hat{H}) \in \Delta_{D,\sigma}.
\]

### 2.4. Unanimity

The next properties we consider capture the idea that if all participants are unanimous with respect to some aspect of the domain, then this unanimity should be reflected in the social outcome. First, the Unanimous attack (UA) condition says that if every participant considers that \( x \) attacks \( x' \), then this attack should be present in the social outcome: \( \gamma \) satisfies the unanimous attack property if:

\[
\forall (\mathcal{H}_1, \ldots, \mathcal{H}_n) \in \Delta^n : \forall \epsilon \in \mathcal{X} \times \mathcal{X} : (\epsilon \in \bigcap_{k=1}^n \mathcal{H}_k \rightarrow \epsilon \in \gamma(\mathcal{H}_1, \ldots, \mathcal{H}_n)).
\]

The \( \sigma \)-unanimity (\( \sigma \)-U) condition says that if \( S \in \mathcal{E}_\sigma \) of every agent’s argument system, then \( S \) should be in \( \mathcal{E}_\sigma \) of the social argument system: \( \gamma \) satisfies the \( \sigma \)-unanimity property if:

\[
\forall \hat{H} \in \Delta^n : \bigcap_{k=1}^n \mathcal{E}_\sigma(\mathcal{H}_k) \subseteq \mathcal{E}_\sigma(\gamma(\mathcal{H}_1, \ldots, \mathcal{H}_n))
\]

The \( ca_\sigma \)-unanimity (\( ca_\sigma \)-U) condition says that every argument that is credulously accepted by all participants should be credulously accepted by the social argument system: \( \gamma \) satisfies the \( ca_\sigma \)-unanimity property if:

\[
\forall (\mathcal{H}_1, \ldots, \mathcal{H}_n) \in \Delta^n : \bigcap_{k=1}^n ca_\sigma(\mathcal{H}_k) \subseteq ca_\sigma(\gamma(\mathcal{H}_1, \ldots, \mathcal{H}_n))
\]

Finally, the \( sa_\sigma \)-unanimity (\( sa_\sigma \)-U) condition says that arguments which are sceptically accepted by all participants should be sceptically accepted in the social argument system:

\[
\forall (\mathcal{H}_1, \ldots, \mathcal{H}_n) \in \Delta^n : \bigcap_{k=1}^n sa_\sigma(\mathcal{H}_k) \subseteq sa_\sigma(\gamma(\mathcal{H}_1, \ldots, \mathcal{H}_n)).
\]
As before, we naturally obtain possibilities: some agent in the social argument system. The output must be one of the positions presented to the procedure as an input. The process must not “have its own opinion” on the arguments. Closure can be understood as a kind of “impartiality” criterion: the argument aggregation it produces something as an output, then this must have been present as an input. Thus, Closure simply says that the argument aggregation function must not “invent” things: if this should be reflected in the social argument system. We can, again, identify several variants of majority. An argument aggregation function \( \gamma \) satisfies the majority attack (MAJ-A) property if it satisfies the following condition:

\[
\forall (H_1, \ldots, H_n) \subseteq \Delta^n : \forall \epsilon \in \mathcal{X} \times \mathcal{X} : \\
\left( \left\{ \{H_i : \epsilon \in H_i\} \right\} > \frac{n}{2} \right) \rightarrow \epsilon \in \gamma(H_1, \ldots, H_n).
\]

We define \( \sigma \)-majority, \( ca_{\sigma} \)-majority, and \( sa_{\sigma} \)-majority (denoted respectively, \( \sigma \)-MAJ, \( ca_{\sigma} \)-MAJ, \( sa_{\sigma} \)-MAJ) via

\[
\forall (H_1, \ldots, H_n) \subseteq \Delta^n : \forall S \subseteq \mathcal{X} : \\
\left( \left\{ \{H_i : S \in E_{\sigma}(H_i)\} \right\} > \frac{n}{2} \right) \rightarrow S \in E_{\sigma}(\gamma(H_1, \ldots, H_n))
\]

\[
\forall (H_1, \ldots, H_n) \subseteq \Delta^n : \forall x \in \mathcal{X} : \\
\left( \left\{ \{H_i : x \in ca_{\sigma}(H_i)\} \right\} > \frac{n}{2} \right) \rightarrow x \in ca_{\sigma}(\gamma(H_1, \ldots, H_n))
\]

\[
\forall (H_1, \ldots, H_n) \subseteq \Delta^n : \forall x \in \mathcal{X} : \\
\left( \left\{ \{H_i : x \in sa_{\sigma}(H_i)\} \right\} > \frac{n}{2} \right) \rightarrow x \in sa_{\sigma}(\gamma(H_1, \ldots, H_n))
\]

### 2.5. Majority

Unanimity properties seem intuitively reasonable: if we all agree on something, then this should be realised in the social argument system. They also, however, seem in some sense unlikely, since they require a complete consensus. A weaker, but arguably still reasonable property, is that of majority: if a majority of agents agree on something, then this should be reflected in the social argument system. We can, again, identify several variants of majority. An argument aggregation function \( \gamma \) satisfies the majority attack (MAJ-A) property if it satisfies the following condition:

\[
\forall (H_1, \ldots, H_n) \subseteq \Delta^n : \forall \epsilon \in \mathcal{X} \times \mathcal{X} : \\
\left( \left\{ \{H_i : \epsilon \in H_i\} \right\} > \frac{n}{2} \right) \rightarrow \epsilon \in \gamma(H_1, \ldots, H_n).
\]

We define \( \sigma \)-majority, \( ca_{\sigma} \)-majority, and \( sa_{\sigma} \)-majority (denoted respectively, \( \sigma \)-MAJ, \( ca_{\sigma} \)-MAJ, \( sa_{\sigma} \)-MAJ) via

\[
\forall (H_1, \ldots, H_n) \subseteq \Delta^n : \forall S \subseteq \mathcal{X} : \\
\left( \left\{ \{H_i : S \in E_{\sigma}(H_i)\} \right\} > \frac{n}{2} \right) \rightarrow S \in E_{\sigma}(\gamma(H_1, \ldots, H_n))
\]

\[
\forall (H_1, \ldots, H_n) \subseteq \Delta^n : \forall x \in \mathcal{X} : \\
\left( \left\{ \{H_i : x \in ca_{\sigma}(H_i)\} \right\} > \frac{n}{2} \right) \rightarrow x \in ca_{\sigma}(\gamma(H_1, \ldots, H_n))
\]

\[
\forall (H_1, \ldots, H_n) \subseteq \Delta^n : \forall x \in \mathcal{X} : \\
\left( \left\{ \{H_i : x \in sa_{\sigma}(H_i)\} \right\} > \frac{n}{2} \right) \rightarrow x \in sa_{\sigma}(\gamma(H_1, \ldots, H_n))
\]

### 2.6. Closure

Closure simply says that the argument aggregation function must not “invent” things: if it produces something as an output, then this must have been present as an input. Thus, closure can be understood as a kind of “impartiality” criterion: the argument aggregation process must not “have its own opinion” on the arguments.

The first closure property considered (CLO) says that the social position produced as output must be one of the positions presented to the procedure as an input:

\[
\forall (H_1, \ldots, H_n) \subseteq \Delta^n : \exists i \in \mathcal{N} : \gamma(H_1, \ldots, H_n) = H_i.
\]

The attack closure (AC) property requires that if \((x, x') \in \mathcal{H}^\ast\) then \((x, x')\) must be in \(\mathcal{H}_i\) for some agent \(i \in \mathcal{N}\): \(\gamma\) is attack closed if:

\[
\forall (H_1, \ldots, H_n) \subseteq \Delta^n : \gamma(H_1, \ldots, H_n) \subseteq (H_1 \cup \cdots \cup H_n).
\]

The \(\sigma\)-closure property (\(\sigma-C\)) says that if a set of arguments \(S\) is considered a possibility in the social argument system \(\mathcal{H}^\ast\), then \(S\) must have been considered a possibility by some agent \(i\). Thus \(\sigma-C\) says that an argument aggregation function cannot “invent” possibilities: \(\gamma\) is said to be \(\sigma\)-closed if:

\[
\forall (H_1, \ldots, H_n) \subseteq \Delta^n : \forall S \in E_{\sigma}(\gamma(H_1, \ldots, H_n)) : S \in \bigcup_{k=1}^{n} E_{\sigma}(H_k)
\]

As before, we naturally obtain \(ca_{\sigma}\)-closure (\(ca_{\sigma}-C\)) and \(sa_{\sigma}\)-closure (\(sa_{\sigma}-C\)) as,
∀(H₁, ..., Hₙ) ∈ Δⁿ : ∀x ∈ caₜ(H₁, ..., Hₙ) : x ∈ ∪ₖ=1ⁿ caₜ(Hₖ)

∀(H₁, ..., Hₙ) ∈ Δⁿ : ∀x ∈ saₜ(H₁, ..., Hₙ) : x ∈ ∪ₖ=1ⁿ saₜ(Hₖ)

2.7. Context Independence

The mechanisms introduced above define an aggregation function to have signature

γ : Δⁿ → Δ

The domain Δⁿ allows the possibility that ⟨i, j⟩ is an attack in γ(H₁, ..., Hₙ) to be conditioned by the presence/absence of “unrelated” attacks in ⟨H₁, ..., Hₙ⟩: this may happen even with anonymous, attack closed functions, e.g.

⟨x₁, x₂⟩ ∈ γ(H₁, ..., Hₙ) ⇔ \bigveeₖ=1ⁿ ((x₁, x₂) ∈ Hₖ) \land \bigveeₖ=1ⁿ ((x₁⁶, x₁⁵) ∈ Hₖ)

We can avoid this by defining γ as a |X × X| tuple

γ = (γ₁,₁, γ₁,₂, ..., γ₁,j, ..., γ|X|,|X|)

where γᵢ,j : ⟨⊤, ⊥⟩ⁿ → ⟨⊤, ⊥⟩ describes the presence or absence of the attack ⟨i, j⟩ solely in terms of the presence or absence of the corresponding attacks in Hₖ, i.e. γᵢ,j is a propositional function of n variables whose values are determined by ⟨i, j⟩ ∈ Hₖ, i.e.

γᵢ,j(y₁, y₂, ..., yₙ) where yₖ = ⊤ ⇔ ⟨i, j⟩ is an attack in Hₖ

We say that γ is context independent, if there exists a collection

γ = (γ₁,₁, γ₁,₂, ..., γ₁,|X|, ..., γ|X|,|X|)

of |X × X| propositional functions such that for all \( \hat{H} \in Δ^n \) the Boolean value \( ⟨i, j⟩ \in γ(\hat{H}) \) is equal to \( γᵢ,j(aᵢ,j₁, ..., aᵢ,jₙ) \) where \( aᵢ,jₖ = ⊤ \) if and only \( ⟨i, j⟩ \in Hₖ \).

The properties considered are still well defined using context independent aggregation functions.

**Theorem 1** Let γ be a context independent aggregation function. If γ satisfies anonymity then γ : Δⁿ → Δ is computable in NC¹, i.e. by an \( O(\log |X|) \) depth, polynomial size propositional formula.

**Proof:** (Outline) Any γ which is both anonymous and context-independent can be expressed as a collection of symmetric Boolean functions. As a result, computing γ only involves “unary-to-binary” addition (and some polynomial additional computation).
3. Complexity of Aggregation Function Analysis

Prior to dealing with how hard it is to determine whether a given argument aggregation function has some property, we need to describe how such functions are presented as instances. Notice that formalisms based on Turing machine or similar encodings of function behaviour will create issues concerning verifying that the specified program delivers its output within a “reasonable” time or, indeed, whether it terminates at all. We avoid such problems in the representation formalism now described.

Observe that any aggregation function $\gamma(H_1,\ldots,H_n)$ corresponds to a collection of $|X| \times |X|$ propositional functions $\gamma_{i,j}$ each dependent on $n \times |X| \times |X|$ propositional variables encoding the attacks present in the contributing systems. We can thus use schemes providing general representation methods for propositional logic functions, e.g. Boolean circuits, i.e. straight-line programs (loop-free) (SLPs) or propositional formulae over some specified (complete) basis, e.g. $\{\land,\lor,\neg\}$, as an encoding method. In particular as a basis for considering the complexity of determining whether a given aggregation function has a specific property of interest. As an example, consider Anonymity: (ANON)

**Instance:** $\gamma : \Delta^n \rightarrow \Delta$, an argument aggregation function.

**Question:** Is it the case that for all $\hat{H} \in \Delta^n$ and $\hat{H}' \in \Pi(\hat{H})$ we have $\gamma(\hat{H}) = \gamma(\hat{H}')$?

Adopting this convention, we note that anonymous aggregation functions which are not context independent are easily shown to require significant size representations, via standard techniques, cf. [7, p. 273].

For decision problems such as ANON we assume instances are presented using some reasonable representation for propositional logic functions such as those outlined above.

**Theorem 2** ANON is $\text{co-NP}$–complete.

**Proof:** To establish $\text{ANON} \in \text{co-NP}$ consider the complementary problem – co-ANON – which reports true for a given instance $\gamma$ whenever there is some $\hat{H} \in \Delta^n$ and $\hat{H}' \in \Pi(\hat{H})$ for which $\gamma(\hat{H}) \neq \gamma(\hat{H}')$. That this problem is in $\text{NP}$ is easily seen using an algorithm that guesses $\hat{H} \in \Delta^n$ and $\hat{H}' \in \Pi(\hat{H})$. Regardless of the specific propositional encoding scheme used for $\gamma$, one can check in polynomial time (in the size of this scheme) whether $\gamma(\hat{H}) \neq \gamma(\hat{H}')$.

To show that ANON is $\text{co-NP}$–hard, we use a reduction from UNSAT, without loss of generality limited to 3-CNF formulae. Suppose that $\varphi(Z_n) = \land_{i=1}^{m} C_i$ is an instance of UNSAT over propositional variables $Z_n = \{z_1, z_2, \ldots, z_n\}$ with each $C_i = y_{i,1} \lor y_{i,2} \lor y_{i,3}$ so that $y_{i,j}$ is a literal ($z$ or $\neg z$) over some variable $z \in Z_n$. Without loss of generality we may assume that no clause contains a literal, $y$, and its complement $\neg y$.

We construct an argument aggregation function, $\Gamma_{\varphi}$ which will be anonymous if and only if $\varphi$ is unsatisfiable.

$\Gamma_{\varphi}$ will aggregate $2n$ systems $(H_1, H_2, \ldots, H_{2n})$, each $H_i$ being defined as a system of $2n$ arguments $(x_{1}, x_{2}, \ldots, x_{2n-1}, x_{2n})$. Such a collection $\hat{H} \in \Delta^{2n}$ is called a legitimate input if it satisfies all of the following:

- **L1.** The attack $(x_{2i-1}, x_{2i})$ is present in exactly one system $H \in \{H_1, \ldots, H_{2n}\}$.
- **L2.** The attack $(x_{2i}, x_{2i-1})$ is present in exactly one system $H \in \{H_1, \ldots, H_{2n}\}$.
If $H_k$ is presented as the $(2n) \times (2n)$ adjacency matrix $[a_{i,j}^k]$ (s.t. $a_{i,j}^k = \top$ iff $(i,j)$ is an attack in $H_k$), then the conditions are easily tested using a polynomial size formula: (L1) and (L2) are, respectively, captured by

$$
\bigwedge_{i=1}^{2n} \left( \bigvee_{k=1}^{2n} a_{2i-1,2i}^k \right) \land \bigwedge_{k=1}^{2n} \left( \neg a_{2i-1,2i}^k \lor \neg a_{2i,2i-1}^k \right)
$$

$$
\bigwedge_{i=1}^{2n} \left( \bigvee_{k=1}^{2n} a_{2i,2i-1}^k \right) \land \bigwedge_{k=1}^{2n} \left( \neg a_{2i,2i-1}^k \lor \neg a_{2i-1,2i-1}^k \right)
$$

Let $[g_{i,j}](\hat{H})$ with $1 \leq i,j \leq 2n$ denote the system that is output by $\Gamma_\varphi$. We first ensure that the behaviour of $\Gamma_\varphi$ is anonymous whenever its input fails to be legitimate by setting

$$\hat{H} \text{ is not legitimate } \implies \forall 1 \leq i,j \leq 2n \ g_{i,j} = \bot$$

Notice that if $\hat{H}$ fails to be legitimate then every $\hat{H}' \in \Pi(\hat{H})$ will fail to be so. Now assuming $\hat{H}$ is a legitimate collection of systems to be aggregated by $\Gamma_\varphi$. We can define an assignment to $\alpha(\hat{H}') \langle z_1, \ldots, z_n \rangle$ for each $\hat{H}' \in \Pi(\hat{H})$ as follows:

- If $a_{2i-1,2i}^k = \top$ and $a_{2i,2i-1}^j = \bot \forall 1 \leq j < k$ then $z_i = \bot$ in $\alpha(\hat{H}')$.
- If $a_{2i,2i-1}^j = \top$ and $a_{2i-1,2i}^k = \bot \forall 1 \leq j < k$ then $z_i = \top$ in $\alpha(\hat{H}')$.

We are now ready to describe the operation of the aggregation function $\Gamma_\varphi$ in full, as Alg. 1

It is clear that Alg. 1 describes a polynomial time computation and can, thus, be effected by a polynomial size SLP.\footnote{In fact, Alg. 1 may be implemented by a polynomial size formula.}

First suppose that $\varphi(Z_n)$ is unsatisfiable and consider any $\hat{H} \in \Delta 2^n$ given as input to $\Gamma_\varphi$. If $\hat{H}$ is not a legitimate input then certainly every $\hat{H}' \in \Pi(\hat{H})$ will fail to be legitimate and thus the output of $\Gamma_\varphi(\hat{H})$ will be unchanged over all $\hat{H}' \in \Pi(\hat{H})$. If $\hat{H}$ is a legitimate input then the output of $\Gamma_\varphi$ will differ for distinct permutations in $\Pi(\hat{H})$ only if there are assignments, $\alpha$ and $\beta$ to $Z_n$ for which $\varphi(\alpha) \neq \varphi(\beta)$. From the assumption that

```plaintext
Algorithm 1 The Aggregation Function $\Gamma_\varphi$
1: if $\hat{H}$ is not legitimate then
2: $g_{i,j} = \bot$ for all $1 \leq i,j \leq 2n$
3: else
4: Construct the assignment $\alpha(\hat{H})$ to $Z_n$ defined by the input ordering $\hat{H}$.
5: if $\alpha(\hat{H})$ does not satisfy $\varphi(Z_n)$ then
6: $g_{i,j} = \bot$ for all $1 \leq i,j \leq 2n$
7: else
8: $g_{i,j} = \top$ if and only if $z_k = \bot$ and $(i,j) = (2k-1,2k)$
9: end if
10: end if
```
\( \varphi(Z_n) \) is unsatisfiable this cannot happen.\(^2\) We deduce that if \( \varphi \) is unsatisfiable then \( \Gamma_\varphi \) is anonymous. On the other hand suppose that \( \Gamma_\varphi \) is anonymous. Then from I.6 of Alg. 1 the output from \( \Gamma_\varphi(H') \) will be unchanged over all \( H' \in \Pi(H) \). Since every assignment to \( Z_n \) will result from some \( H' \in \Pi(H) \) it follows that if \( \varphi(\alpha) = \bot \) for at least one assignment to \( Z_n \) then \( \varphi(\alpha) \) will be \( \bot \) for every assignment. Since \( \varphi \), trivially, has at least one unsatisfying assignment we deduce from \( \Gamma_\varphi \) being anonymous that \( \varphi(Z_n) \) is unsatisfiable.

By a similar approach it can be shown that \( \text{UA, MAJ-A and AC} \) are \( \text{coNP–complete} \). The approach of associating a particular assignment to propositional variables depending on the ordering of systems in \( \mathcal{H} \) can also be used to derive complexity bounds in classes other than \( \text{coNP} \).

**Theorem 3** pe-\( \text{SNT} \), the problem of deciding given an argumentation aggregation function, \( \gamma \), whether \( \gamma \) is pe strongly non-trivial is \( \Pi^p_2–\text{complete} \).

**Proof:** We focus on \( \Pi^p_2–\text{hardness} \), omitting the upper bound for space reasons. We reduce from \( \text{QSAT}^{\Pi^p_2}_2 \), the problem of deciding for a given 3-CNF formula \( \varphi(Y_n, Z_n) \) over two disjoint sets of propositional variables, whether

\[
\forall \alpha_Y \in \{\top, \bot\}^n \exists \beta_Z \in \{\top, \bot\}^n \; \varphi(\alpha_Y, \beta_Z) = \top
\]

Given \( \varphi(Y_n, Z_n) = \land_{j=1}^m C_j \) each \( C_j \) a conjunction of three literals over \( Y_n \cup Z_n \), \( \Gamma_\varphi \) aggregates \( 2n \) systems \( \langle \mathcal{H}_1, \ldots, \mathcal{H}_{2n} \rangle \) each system \( \langle X_\varphi, A^{(k)}_\varphi \rangle \) having \( 4n + m + 1 \) arguments

\[
X_\varphi = \{x_1, \ldots, x_{2n}, x_{2n+1}, \ldots, x_{4n}, x_{4n+1}, \ldots, x_{4n+m}, x_{4n+m+1} \}
\]

We say \( \mathcal{H} = \langle \mathcal{H}_1, \ldots, \mathcal{H}_{2n} \rangle \) is legitimate if \( \mathcal{H} \) satisfies conditions (L1) and (L2) from the proof of Thm. 2 with respect to the variables \( \langle x_1, \ldots, x_{2n} \rangle \), and \( |A^{(k)}_\varphi| = 1 \) for every \( 1 \leq k \leq 2n \). We omit the full description of the algorithm computing \( \Gamma_\varphi \). In informal terms its actions are: if the input collection, \( \mathcal{H} \) fails to be legitimate then the output system, \( \Gamma_\varphi(\mathcal{H}) \) has no attacks present: such a system is clearly pe non-trivial. If, on the other hand, \( \mathcal{H} \) defines a legitimate input corresponding to some assignment \( \alpha_Y \) to \( Y_n \) (as described by the ordering of individual systems in \( \mathcal{H} \)), then the argumentation system \( \Gamma_\varphi \) corresponds to the so-called standard translation from 3-CNF to argumentation systems, [13, p.91], modified so that every \( y_i, \neg y_i \), and \( C_j \) argument (those in the range \( 1 \leq i \leq 2n, 4n + 1 \leq i \leq 4n + m \)) is self-attacking (and thus cannot form part of a preferred extension). In addition the argument \( \varphi_i \), i.e. \( x_{4n+m+1} \) attacks each of the \( x_i \) literals, i.e. \( x_j \) with \( 2n + 1 \leq j \leq 4n \). The actual CNF translated is \( \varphi(\alpha_Y, Z_n) \). For space reasons we, again, omit the details of the proof showing \( \varphi(Y_n, Z_n) \) is accepted as an instance of \( \text{QSAT}^{\Pi^p_2}_2 \) iff \( \Gamma_\varphi(\mathcal{H}) \) is pe non-trivial.

Similar arguments may be used to prove pe–\( \text{wnt} \), \( \sigma-\text{SNT} \) and \( \sigma-\text{WNT} \) to be \( \Pi^p_2–\text{complete} \) (when \( \sigma \in \{ \text{adm}, st \} \)) and \( \text{GND-SNT} \), \( \text{GND-WNT} \) both \( \text{coNP–complete} \).

\(^2\)Notice that since \( \varphi \) is in \text{CNF} with a literal and its negation not occurring in the same clause, it is always possible to choose an assignment that fails to satisfy \( \varphi \).
4. Some Impossibility Results

An important body of work in the study of preference and judgment aggregation deals with so-called “impossibility theorems”: that is, sets of properties that aggregation functions cannot be guaranteed to satisfy simultaneously. Given the connection of our model with these areas, one would expect similar results with respect to argument aggregation. That this expectation is justified is shown in the very basic result below.

Lemma 1

a. Any argument aggregation procedure satisfying \( \text{MAJ-A} \) and \( \text{AC} \) cannot satisfy any of \( \text{SNT} \), \( \text{WNT} \), \( \text{SD} \), \( \text{WD} \), \( \text{CLO} \), \( \text{pe-C} \), \( \text{ca-C} \), \( \text{sa-C} \).

b. There is no argument aggregation function that simultaneously satisfies \( \text{MAJ-A} \), context independence and \( \sigma\text{-WD} \) (for any choice of \( \sigma \) considered above).

Proof: Both parts follow by considering the situation where we have three agents \( \mathcal{N} = \{1, 2, 3\} \), and

\[
\mathcal{H}_1 = \{(x, y, z), ((x, y), (y, z))\}
\]

\[
\mathcal{H}_2 = \{(x, y, z), ((y, z), (z, x))\}
\]

\[
\mathcal{H}_3 = \{(x, y, z), ((z, x), (x, y))\}
\]

each agent is associated with a decisive (and hence non-trivial) argument system. We have \( \text{pe}(\mathcal{H}_1) = \{(x, z)\} \), \( \text{pe}(\mathcal{H}_2) = \{(x, y)\} \), and \( \text{pe}(\mathcal{H}_3) = \{(y, z)\} \). Now, each attack \( (x, y), (y, z), (z, x) \) is supported by a majority of agents thus, all three attacks will appear in \( \mathcal{H}^* \): \( \mathcal{H}^* \) is not decisive; in fact, it is trivial: \( \text{pe}(\mathcal{H}^*) = \{\emptyset\} \). From this it immediately follows that a procedure satisfying \( \text{MAJ-A} \) and \( \text{AC} \) cannot satisfy \( \text{SNT} \), \( \text{WNT} \), \( \text{SD} \), \( \text{WD} \), \( \text{CLO} \), \( \text{pe-C} \), \( \text{ca-C} \), \( \text{sa-C} \).

5. Related Work

The topic with which this paper has been concerned has similarities to a number of issues considered in earlier work. In particular that on belief merging and social choice, see e.g., the overview of [10] and work of [12,8,9]; preference and judgment aggregation, e.g., [5,11]. Probably most directly related to the topic of the current paper is the work of Coste-Marquis et al. [4] in which the question of reporting a “sensible” unified framework given a collection of \( k \) frameworks is examined. The results of [4] highlight the inadequacies of naive approaches – such as majority voting or forming the union/intersection of attack sets – and present a formulation of aggregation function. Unlike our approach, the “aggregation functions” of [4] are \textit{quantitative} mechanisms aimed at capturing the extent to which two frameworks differ and thus serve as a device to exploit in \textit{merging} frameworks rather a a means of reporting a single \( \text{AF} \) directly. While some algorithmic issues are discussed in [4], issues regarding computational complexity are not pursued in depth. The notion of so-called \textit{merging operators} and their properties has featured in a number of studies of belief merging: one aspect of interest – not considered directly in our development above – being the susceptibility of such operations to strategic manipulation, e.g., as examined in Everaere et al. [8,9] and Konieczny et al. [12]. It may be observed that argumentation based treatments of preferences provide
a vehicle by which agents are able to justify individual preferences (and, implicitly, identify grounds for attacking preferences of other agents). Thus preference and judgment aggregation – to which a common treatment has been posited by Grossi [11] may potentially be examined in the context of argumentation aggregation.

6. Conclusions and Development

Social choice theory considers the problem of preference aggregation: how the potentially conflicting preferences of agents in a society can be combined to obtain a social outcome, in such a way as to reflect as faithfully as possible the preferences of the individual agents within the society. Two key research issues in social choice theory are the axiomatization of desirable properties of social choice procedures, and the extent to which preference aggregation procedures can and do satisfy these axioms. In the argument aggregation problem, we are concerned not with aggregating preferences but with aggregating abstract argument systems. In this paper, we have considered two issues: possible axioms for argument aggregation, and the possibility of verifying whether particular argument aggregation procedures do or do not satisfy these axioms, for a particular computational model of argument aggregation procedures. There are many issues for future work. First, one might consider possible extensions to the axioms, and associated possibility/impossibility results. In particular, it would be of interest to explore maximal sets of conditions which are realisable by appropriate aggregation functions.

References