Faculty of Natural and Mathematical Sciences
Department of Informatics

King's College London
Strand Campus, London,
United Kingdom

7CCSMPRJ

Individual Project Submission 2022/23

Name:	William Feeney
Student Number:	21144138
Degree Programme:	MSc Artificial Intelligence
Project Title:	System To Support A Group Of Humans In Making Joint Decisions With Arguments
Supervisor:	Dr Elizabeth Black
Word Count:	14,627

RELEASE OF PROJECT

Following the submission of your project, the Department would like to make it publicly available via the library electronic resources. You will retain copyright of the project.

☑ I **agree** to the release of my project

☐ I **do not** agree to the release of my project

Signature: 

Date: August 11, 2023

Department of Informatics
King's College London
United Kingdom

7CCSMPRJ Individual Project

System To Support A Group Of Humans In Making Joint Decisions With Arguments

---

Name: **William Feeney**
Student Number: 21144138
Course: MSc Artificial Intelligence

Supervisor: Dr Elizabeth Black

This dissertation is submitted for the degree of MSc in MSc Artificial Intelligence.

Abstract

This overall aim of this project is to design a system to support a group of humans in making a joint decision through arguments, taking into account their subjective preferences. The system helps users in situations where they have to make a decision for or against a particular choice or course of action, but beyond that the system's use is not limited to any particular type of situation. This project considers how to determine this optimal decision from a given set of arguments and subjective user input. Modern argumentation theory provides a variety of different frameworks for modelling arguments and the relationships between them, and therefore underpins the methodology used by the system. Given the vast number of frameworks available I consider three different systems that are constructed from combining and extending existing frameworks, each requiring a different form of subjective input.

In particular, the key contributions of this project are to determine: i) a methodology to evaluate existing unipolar preference-based and value-based argumentation frameworks within a bipolar setting, ii) the most appropriate way to aggregate the views of multiple users in these two frameworks, iii) how to extend existing voting frameworks to allow for weighted votes, and iv) how well these frameworks perform with actual users.

The project evaluates the three systems against a set of desired specifications and against feedback provided by participants in user testing. This is used to determine whether any of these systems meets the overall project aim, and whether it is possible to determine if one of the systems is superior to the others.

Acknowledgements

I would first like to thank my project supervisor Dr Elizabeth Black for her support and feedback in managing the project and writing this report. I would also like to thank my participants for taking part in the user testing, and taking time to provide their very useful feedback. Finally, I would like to thank my partner for her continued encouragement despite having to endure several months of me rambling on about abstract argumentation frameworks.

Contents

List of Figures

1 Introduction

1.1 Motivation

History provides us with an abundance of experiments in group political decision making. The Athenians are the typical example of an early foray into democracy. From the 6th century BC, you had a say in the running of the Athenian state as long as you were in the estimated 10%-20% of the population who was a free, adult male whose parents were both Athenian citizens (originally just an Athenian father had been enough but Pericles tightened up this rule in 451 BC) [44].

Conceived around the same time, the senate of the Roman Republic had a more oligarchical flavour with membership initially based on hereditary privilege. Interestingly, the Roman political structure also allowed for the appointment of a single dictator during times of crisis, often invasions by the Carthaginians, in which the dictator temporarily ruled with complete control. Clearly in crunch times the Romans' faith in group decisions fell by the wayside. In 44 BC, when Julius Caesar famously declared himself *"dictator perpetuo (dictator for life)"*, perhaps it had always been inevitable that the senate's power would gradually dwindle in favour of an all-powerful emperor [47].

Nowadays, most developed nations are governed by some form of democracy in which elected representatives are responsible for making decisions on behalf of the wider population. These decisions are informed by debates in which members of different political parties construct arguments for and against possible options. Whilst much can be said about the current 'post-truth' era and influence of fake-news, it is clear that the success of these arguments is not determined solely by their logical soundness. Rather there are a myriad of factors at play, including the subjective preferences of the audience and their various biases. Individuals are influenced by irrelevant information [25] and their own extraneous circumstances (for example hungry judges are less lenient) [16]. Unsurprisingly, determining the right decision following a debate isn't always straightforward.

Therefore, a key motivation for this project is the way in which democratic governments make decisions, for example designing new policies or agreeing new laws, whilst taking into account the views of the population that they are governing. Specifically, I consider how to design a system to aid this type of decision-making process. It is a system where the strength of an argument is not determined only by its logical merits, but also its subjective appeal. For various social and ethical reasons, I would not envision my system to be a replacement for human decision-makers but to be used as a supporting tool to enable better decision-making.

1.2 Argumentation Theory

Given humanity's history of conflict it is unsurprising that the origins of argumentation theory stretch back a long way. Whilst my project focuses on the theories that have been developed from the end of the 20th century onwards, it was the ancient Greeks that set

the foundations of argumentation theory with their study of logic and epistemology [51]. Plato was a key proponent of the dialectic method and often presented his philosophical arguments in the form of an imaginary debate between Socrates and another individual [36].

Moving forward to the present day, modern argumentation theory retains some of the logical elements familiar to the ancient Greeks, but has been developed for use in more practical situations. This has been achieved by loosening some of the stricter rules of formal logic to better reflect the real world and its inherent uncertainty, enabling the development of a vast number of subjective and probabilistic argumentation frameworks. It has also become an increasingly important field of study within the artificial intelligence community where it is seen as a potential solution to the problem of how to create explainable AI [49].

Whilst there has been a significant amount of time devoted to theoretical research on designing new argumentation frameworks, the study of how well these actually perform in practice appears to be less developed. This project aims to help address that imbalance by designing an argumentation system, using concepts from existing frameworks, and testing it with actual users.

1.3 Problem Discussion

The type of scenario in which my system will be applied is one in which a proposed course of action, the relevant arguments, and the relationships between those arguments (which arguments attack or support which other arguments) have already been set out to the users. The process by which these elements are collected is outside the scope of this project, however I would expect a form of debate in which individuals take turns putting forward their arguments whilst also specifying which other arguments they are attacking or supporting. It may also be the case, that the users are provided with the set of arguments from another group of individuals, possibly subject experts or in the context of political decisions from a debate between politicians.

Alternatively, in a more advanced system, users might only be required to put forward their arguments and the system will automatically determine the relationships between them. This system would likely combine various natural language processing and machine learning techniques. Such a system was developed by Alsinet et al. whilst analysing arguments on Twitter [2]. However, a possible barrier to this approach is the substantial amount of training data required to be able to label new arguments correctly. Alsinet trained four different models using 582 pairs of tweets and achieved only a modest level of labelling accuracy, ranging from 48% to 60%.

Whilst my system can't facilitate the actual debate, it can construct a representation of the debate and use this to determine an outcome, taking into account the subjective preferences of all the users. To do this requires subjective input from each user, which depending on the version of the system used is either a preference ordering of the arguments, a

preference ordering of the values promoted by the arguments or (weighted) votes for or against each argument.

In many cases I would expect the outcome from my system to be similar to the outcome from the users following a group discussion, without any external support. In these instances the system's key benefit will be the increased efficiency of the decision-making process as it should be able to reach an outcome faster than a group discussion. Furthermore, the system should provide a better evaluation of the reasons for why a particular decision was made.

Of course, there will also be situations in which the system's output will differ from that of a group discussion. If a group contains a dominant character then the group's decision may be heavily weighted towards the views of that individual. Or the group may fall prey to the phenomenon of *groupthink* in which the desire for consensus overrides the need to make the best decision. Groupthink has been suggested as the reason behind many major flawed decisions, including Nazi Germany invading the Soviet Union in 1941, NASA launching the Challenger space shuttle and Grunenthal Chemie marketing the drug thalidomide [48]. However, my system should take a more balanced view of the different opinions in the group and avoid these potential biases.

In such instances where my system arrives at a different outcome, the output from the system can be used to analyse the reasons why. I would even argue that it is these situations where the system provides the most value, as it allows users to reassess their decision and address potential flaws in their decision-making process.

1.4 Project Aims

The primary aim of this project is to design a system to help a group of individuals make a joint decision using arguments, taking into account their subjective preferences. In order to achieve this overarching aim, the project has the following sub-aims:

- Using concepts from argumentation theory, consider how the system can effectively represent arguments and the relationships between them.
- Determine how the system will account for the subjective preferences of multiple users when making a decision.
- Determine how the system will produce a final decision.
- Determine what further output the system will produce in order to aid decision decision-making.

2 Literature Review

In the following subsections I review the literature relevant to this project. This includes a discussion of the key features of existing argumentation systems and how they relate to the project aims.

2.1 Modelling Arguments

In designing my system I needed to consider how to represent arguments and the relationships between them. From an AI perspective, the problem of how to create logical models representing arguments has been considered since the mid-1980s. [9] provides an overview of the main research developments during the 1980s and 1990s. In particular, it highlights that a key issue in modelling arguments was how to handle situations where information might be incomplete, uncertain and even contradictory. Classical propositional logic was deemed unsuitable for this problem as it had no means of dealing with these types of issues [43]. Instead the focus shifted to *non-monotonic* logics and their *defeasible* arguments, in which *"conclusions drawn may be later withdrawn when additional information is obtained"* [14]. This type of reasoning was studied extensively by logicians and philosophers, including Pollock who significantly contributed to the field through his work on defeasible reasoning and its use within AI systems [37] [38].

A number of argumentation systems were subsequently created. Pollock developed a system called OSCAR (Online System for Commentary and Response) which implemented his ideas on defeasible reasoning whilst also emphasising the importance of rebuttal and criticism within arguments [37] [39]. However, Vreeswijk argues that in some cases OSCAR can produce false arguments, and that *"these fallacious arguments unjustifiably defeat lines of reasoning that actually should emerge undefeated"* [50]. Another notable approach is Walton's Argumentation Schemes [52], which are a set of predefined templates for constructing arguments in specific contexts, derived from empirical research on real-world arguments. This is a more descriptive approach that for a given situation requires identification of the most appropriate scheme. As such it can have issues generalising to more complicated arguments. These two examples merely scratch the surface of the numerous argumentation systems that have been developed; other systems and their problems are explored in more detail by Vreeswijk [50].

These efforts culminated in Dung's seminal paper on argumentation theory [17]. In this paper, Dung set out his Abstract Argumentation Framework and argued that *"most of the other major approaches to non-monotonic reasoning in AI and logic programming are special forms of [his] theory of argumentation"*. An important aspect of Dung's work is that it is possible to analyse arguments abstractly without needing to have any semantic understanding of their actual content. Another attractive feature is its simplicity - all you need to model an argument is to know what arguments have been put forward and which arguments attack which other arguments. Furthermore, it is straightforward to graphically represent Dung's framework, as demonstrated in Figure 1. The subsequent popularity of

this framework has resulted in many extensions, several of which I will discuss later.

One disadvantage of Dung's framework is that it does not directly allow for supporting arguments, those that are used to strengthen previously made arguments. Both supporting and attacking arguments occur naturally in discussions. For example, I might claim that Thatcher was the best UK Prime Minister of the 20th century. Within Dung's framework you can attack my claim by saying that her policies devastated manufacturing communities. However, I should also be allowed to strengthen my claim by noting that Thatcher oversaw her party's best post-war election results. Therefore a useful extension is the Bipolar Argumentation Framework, which allows for the modelling of supporting arguments [5].

Developing the above example further, you may argue that having strong election results doesn't make someone a good Prime Minister, rather that is determined by the quality of their policies and effectiveness at implementing them. Although we would agree that Thatcher had strong election results, you would be attacking my assertion that this fact supports my initial claim. These types of meta-arguments are not possible to model in Dung's original framework, but are modelled in Extended Argumentation Frameworks [30]. As well as being a useful tool for evaluating if one argument actually has logical implications relating to another argument, these meta-arguments can also be used to express subjective preferences. However, the increased expressiveness of this approach does come with increased complexity.

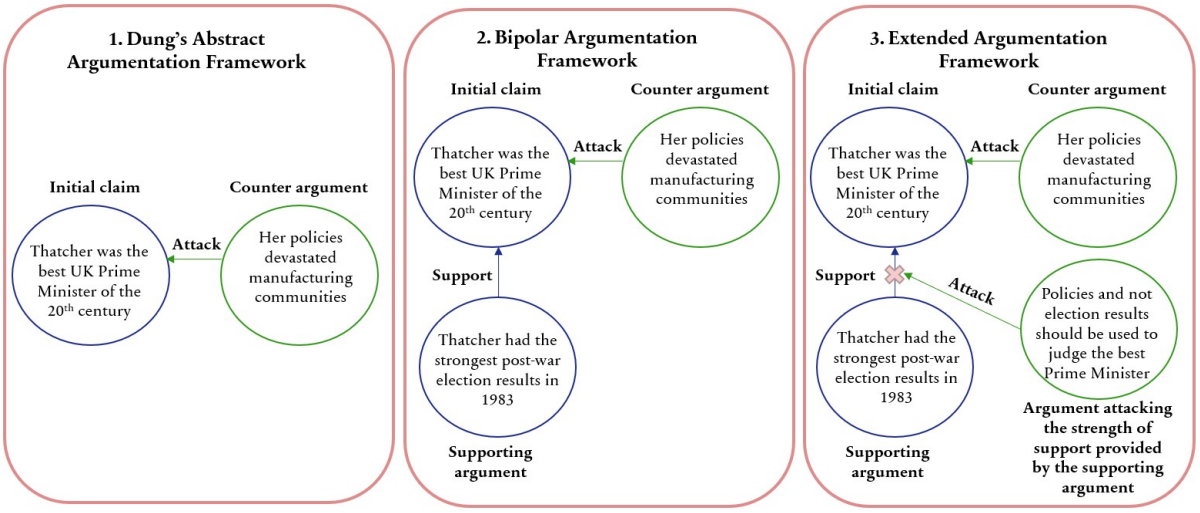


Figure 1: *Examples of increasingly expressive argumentation frameworks.*

Dung also considered what it means for an argument to be acceptable within an argumentation framework. Dung defined different sets of acceptable arguments depending on whether or not they satisfied certain properties. These properties can be used to broadly describe how acceptable an argument is. Dung also showed that for certain types of debates two individuals may believe that different subsets of the arguments are acceptable, with each being able to rationally defend their positions. A challenge for my system is how to determine an overall decision despite such differences between users.

In the following subsection I discuss a likely area of disagreement - the subjective views of individuals.

2.2 Subjective Argumentation

Dung described his framework as adhering to the principle that *"The one who has the last word laughs best"* [17], meaning that in a chain of arguments the last argument will always be the winner. Whilst it may be true for some people that having the last word constitutes winning an argument, typically there will be a range of factors that determines the winner. In particular, an individual's subjective preferences will significantly influence whether they believe one argument is stronger than another.

Amgoud and Cayrol's proposed solution was to extend Dung's framework to allow individuals to assign preferences between arguments, creating a Preference-Based Argumentation Framework [4]. In this framework more preferred arguments can survive attacks from less preferred arguments - the last argument is no longer always the winner.

An alternative approach is the Value-Based Argumentation Framework developed by Bench-Capon [8]. Bench-Capon also extended Dung's work, this time accounting for arguments promoting different social values and the fact that some values will be preferred over others. For example, the abortion debate is driven by the values associated with the right to life and the right to choose. If there was consensus that one of these values was more important than the other then this argument could be resolved by appealing to that value.

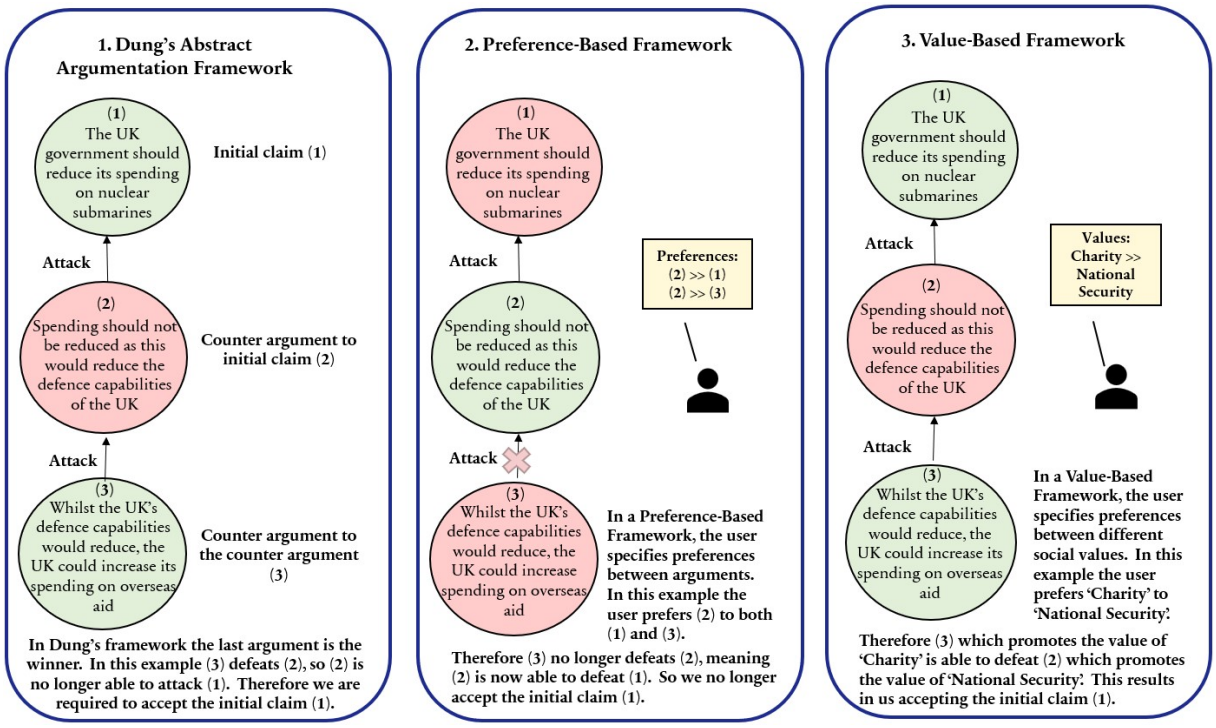


Figure 2: *Examples of subjective argumentation frameworks.*

2.3 Social Argumentation

Another method to incorporate subjectivity is to ask individuals to vote for or against arguments. A key benefit of voting based approaches over the previously discussed frameworks is that they lend themselves well to aggregating the views of multiple individuals.

Leite and Martins designed a Social Abstract Argumentation Framework [28], which integrates social voting into Dung's framework. Specifically, individuals cast votes for and against individual arguments, which are then used to calculate weights for each argument corresponding to their general level of acceptance amongst the group. The system is also consistent with the aims of my project to incorporate the subjective views of multiple users with the authors themselves stating that *"our semantics is not expected to evaluate the logical correctness of arguments, but instead their social acceptance"*. Furthermore, this framework has been extended to also consider how voting can be used to determine the strength of attacks between arguments [19].

Similarly, Rago and Toni developed a Quantitative Argumentation Debate for Voting Framework (QuAD-V) [41], which was motivated by considering how opinion polling could be combined with argumentation. They also developed the concept of a rational voter, and how *dynamic questions* can be used to guide towards irrational voters to a consistent set of views.

2.3.1 Quantitative Argumentation

Relative to preference-based and value-based argumentation, which typically only evaluate to binary outcomes - an argument is either accepted or it isn't - social argumentation lends itself more to quantitative analysis and varying levels of acceptance. For instance, you might agree with two different arguments but find one more convincing than the other. As an example, some might argue you should vote for a certain political party because they have a catchy slogan, whereas others may argue you should vote for them because they have a great track record implementing successful social policies. Hopefully, most would agree the latter argument is stronger (although personally I would find some slogans hard not to vote for - a personal favourite being the Australian Democrats' slogan:*"Keep the B******ds Honest"* [31]).

To this end, various quantitative frameworks have been developed that determine weights or scores for each argument corresponding to their overall level of acceptance. The Social Abstract Argumentation Framework in [28] is one such example, however others include:

- Discontinuity-Free Quantitative Argumentation Debate (DF-QuAD) [42]: a bipolar framework in which arguments have weights, and the final score of each argument is dependent on the weights of its supporters and attackers. The 'Discontinuity-Free' description refers to an improvement upon an earlier version of the framework [7], in which weak supporting or attacking arguments could have a dispro-

portionate impact.

- Weighted Bipolar Argumentation Framework [3]: another bipolar framework with weighted arguments. However, its use of *exponent-based restricted semantics*, an alternative method for calculating the strength of an argument based on its supporters and attackers, has a number of desirable properties not present in other frameworks (discussed further in Section 3.2.3).
- Weighted Argumentation Framework [18]: in contrast to the above two examples, the attacks between arguments are given weights rather than the arguments themselves. This framework also allows the user to set an *inconsistency budget*, which provides a method for reasoning with inconsistent beliefs amongst users.

Having demonstrated the variety of available existing argumentation frameworks, in the next section I outline the desired system specifications, and use these to determine which frameworks to include in the design of my system.

3 Specifications & Design

As a reminder, the overarching goal of the project is to design a system to help a group of individuals make a joint decision using arguments, taking into account the subjective preferences of those individuals. In Section 1, I set out more specific aims and provided clarification on the type of problem my system solves. Taking all this into account, I set out my desired specifications for the system in the following subsection.

3.1 System Specifications

1. The system is required to model arguments and the relationships between them using a representation that the system can then analyse to output a decision based on those arguments.
2. Any modelling restrictions should not result in significant limitations on the situations in which the system can be applied.
3. The system is required to output a single decision based on the arguments and user input given. In any cases where a decision can't be made it should be possible to understand why.
4. The system is required to take into account pre-determined forms of subjective input from each of the users when making a decision. However, any inputs required from the users should be simple in nature and not too onerous for the users to provide.
5. The system is required to take into account the input from all of the users when making a decision. The system should not place greater weight on the input from some users over others.
6. The decisions made by the system should be explainable. This means it should be possible for the users to understand which arguments or factors have influenced the final decision, and ideally to what extent they have influenced that decision. The system should indicate its level of conviction in its decision.
7. The system should be designed in such a way to minimise the risk of individual users from being able to 'game' it ie exploit elements of the system in order to materially increase the likelihood of their preferred outcome being output.

3.2 Comparison of Argumentation Frameworks

In this section I compare the various frameworks discussed in the literature review against the system specifications, explaining how I have selected the most appropriate frameworks for my system. Whilst I considered all specifications at each design stage, I highlight where certain specifications are particularly relevant to a given design choice.

A key result from this section is that each of the frameworks has their own relative

strengths and weaknesses, and from the theory alone I don't believe it is possible to determine an optimal methodology. Therefore, I consider three different options. Whilst it is certainly possible to construct more options, I believe that considering three options ensures the analysis remains manageable whilst allowing for a richer and more interesting evaluation compared to considering only a single approach. It also means I can ensure that each option is different enough to warrant investigating, whilst meeting the constraints of the system specifications.

3.2.1 Modelling Arguments

Key Specifications: 1, 2, 6

A key design choice is which underlying model to use for representing arguments. As discussed in the literature review, a large number of argumentation systems are available. However, I chose Dung's framework [17] as the basis for my system as it avoids the need to have a semantic understanding of the arguments in order to form a representation of them, making it more flexible than other options. Furthermore, its simplicity should help users to understand its output and the vast research on extensions to the framework helps in meeting the other system specifications.

In terms of incorporating extensions to Dung's framework in my system, it is important to get the balance right between having a more expressive model that might better reflect real world arguments against the additional complexity that might impact the ability of users to understand it. The latter point is of genuine concern given that there is mixed evidence regarding the general population's ability to reason and understand arguments. According to Harrell [24], *"many studies suggest that the skills of understanding, evaluating, and producing arguments are generally poor in the population of people who have not had specific training"*. I have therefore purposely taken a simpler approach in line with the specifications that users should be able to understand the output of the system and that any input into the system should be simple.

To this end, I have chosen to use the bipolar argumentation framework extension [5]. I believe that the concept of having both supporting and attacking types of argument is straightforward enough for users to understand and would significantly limit the system otherwise. On the other hand, I believe that the use of more complicated meta-structures, such as in Extended Argumentation Frameworks [30], might confuse some users, especially if they are required to rank or comment on these types of arguments. Therefore, my system avoids explicitly modelling these meta-structures.

3.2.2 Subjective & Social Argumentation Framework Extensions

Key Specifications: 4, 5, 6

In regards to accounting for the subjective views of the users, in the literature review I set out the three following options:

- Preference-Based Frameworks [4], in which users specify their preferences between arguments. The strengths of this approach are that it directly captures the subjective preferences of the users, the required input is simple to understand, and the approach can be used in a wide range of domains. However, individuals may find it difficult to assign preferences between certain arguments. Also the user preferences may not fully capture the moral or ethical dimensions that in some cases should guide decision-making. Finally, there is no agreed consensus for aggregating the different preferences of multiple users or evaluating a bipolar framework with preferences.

- Value-Based Frameworks [8], in which users specify their preferences between different social values relating to the arguments. As users only have to specify preferences between a smaller number of social values, this input should be less onerous to determine than for a Preference-Based Framework. Alternatively, a group of users could agree to use a predefined set of criteria or social values when evaluating the arguments. This approach would also ensure that the appropriate morals and ethics are used in the decision-making. On the other hand, a potential issue when implementing a Value-Based Argumentation Framework is the need to specify which social values correspond to which arguments. There might also be scenarios where if an individual knew that certain social values were highly regarded and so likely to win an argument they may try to link their argument to one of these social values no matter how tenuous the link. For instance, one UK MP recently argued against 20 minute neighbourhoods (the idea that towns and cities should be designed so that the majority of things a resident might need on a day to day basis are within a 20 minute walk or cycle of their home) by calling the idea an *"international socialist concept"* that will *"take away your personal freedoms"* [21]. Even those who hold the social value of personal freedoms in high regard (or socialist values in low regard) may struggle to support this line of reasoning. As with Preference-Based Frameworks there is no agreed consensus on how to evaluate a bipolar value-based framework or aggregate the results of multiple users.

- Voting-Based Frameworks [28] [41], in which users vote (possibly with different weights) for or against each argument. A key strength of this approach is that it is specifically designed to aggregate the views of multiple users, and often in a quantitative way that can provide insights into the overall acceptance of arguments. As such it can be integrated well with other quantitative frameworks. However, possible weaknesses are that as individuals are not required to provide preferences between arguments it can be difficult to assess if there is a winning argument - consider an example argument in which a user votes 'for' every argument. Tactical voting can also cause issues. Similar to Preference-Based Frameworks user votes may also not reflect relevant moral or ethical considerations.

Having considered each approach, I do not believe there is an obvious optimal subjective framework. Therefore, I believe it is worthwhile exploring them all, and so I have

designed a methodology for each of the three types of frameworks: preference-based, value-based and voting-based.

3.2.3 Quantitative Framework Extensions

Key Specifications: 5, 6, 7

There are a variety of quantitative extensions that can be used to evaluate a voting-based framework. Some of these I have ruled out using because they are incompatible with other design choices I have made. For instance, the Social Abstract Argumentation Framework in [28] is only designed to evaluate unipolar rather than bipolar argument frameworks. The Weighted Argumentation Framework in [18] would require users to vote on the strength of attacks between arguments rather than the arguments themselves, which I discounted for similar reasons as the Extended Argumentation Frameworks in [30].

The two remaining evaluation methods I have considered are DF-QuAD from [42] and the Weighted Bipolar Argumentation Framework, specifically the exponent-based restricted semantics, from [3]. Both work in a bipolar framework and have various desirable properties under certain restrictions, namely that frameworks have to be acyclic. For example, these properties include that the strength of an argument is determined only by its basic strength (as determined by the votes) and its supporters and attackers, and that an argument is stronger the less it is attacked and the more it is supported. However, as shown in [3] there are a number of properties not satisfied by DF-QuAD that the exponent-based restricted semantics do satisfy. In particular, an issue with DF-QuAD is that a very weak argument can become very strong due to only a single supporter. This is not possible if using exponent-based restricted semantics, although a very weak argument can still become strong through the combined effect of many supporters. I have therefore chosen to incorporate the exponent-based restricted semantics within my voting system as I believe this is a more robust option and should help reduce any gaming of the system.

A summary of the frameworks I have chosen to use in my systems is given in Figure 3.

<table>
<tr><th></th><th>Preference Based System</th><th>Value Based System</th><th>Voting Based System</th></tr>
<tr><td>Representing arguments</td><td colspan="3">Bipolar Argumentation Framework
(Amgoud, Cayrol & Lagasquie-Schiex)</td></tr>
<tr><td>Evaluating arguments</td><td>Preference-Based Argumentation Framework (Amgoud & Cayrol)</td><td>Value-Based Argumentation Framework (Bench-Capon)</td><td>Exponent-Based Restricted Semantics (Amgoud & Ben-Naim)</td></tr>
<tr><td>Aggregating multiple users</td><td colspan="2">Discussed in Section 4</td><td>Social Abstract Argumentation Framework (Leite & Martins)

QuAD-V Framework (Rago & Toni)</td></tr>
</table>

Figure 3: *A summary of the argumentation frameworks used in my systems.*

4 Methodology & Implementation

Having set out my reasoning for creating three different systems and the rationale for incorporating elements of existing argumentation frameworks into them, in this section I explain the methodology for each system. As set out in Figure 4, at a high-level all of the systems share a similar methodology:

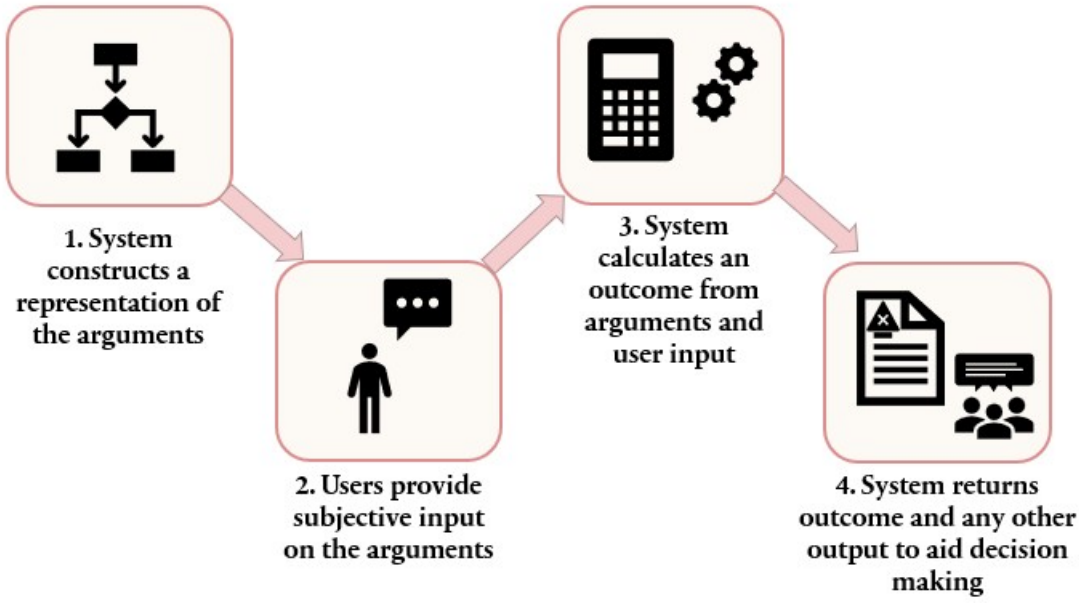


Figure 4: *A high-level summary of the methodology for each of the systems.*

4.1 Abstract Argumentation Frameworks

In each system, the first step is to create a Bipolar Argumentation Framework from a given set of arguments. This framework extends Dung's original Abstract Argumentation Framework (AF), which is defined below. Using this definition the actual content of the arguments does not need to be considered, instead it is the relationships between the arguments that are important.

Definition (Abstract Argumentation Framework): An abstract argumentation framework (AF) is a pair $<AR, R>$ where AR is a set of arguments, and R is a binary relation on AR, ie $R \subseteq AR \times AR$. For two arguments A, $B \in AR$, the meaning of $R(A,B)$ is that A represents an attack against B.

As an example, consider an AF = <{A, B, C, D, E, F}, {(B, A), (C, B), (D, B), (E, F), (F, E)}> which can be represented graphically as in Figure 5. An arrow from one node to another represents an attack. Note that it is possible for two arguments to attack each other as in the case of E and F. [17]

As we now have a way to represent arguments the next challenge is to determine which of those arguments we should accept. According to Dung, a rational individual will consider an argument A to be *acceptable* if the agent can defend A against all attacks on

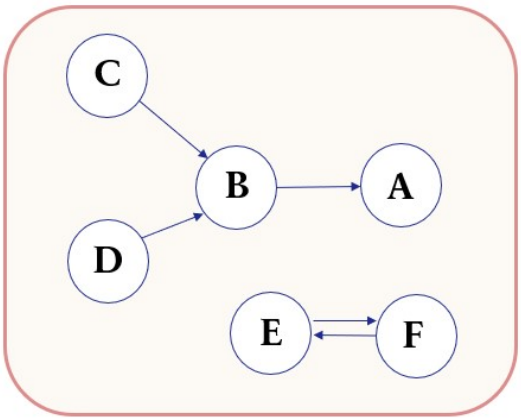


Figure 5: *A graphical representation of an argumentation framework.*

A using the arguments available to them. Therefore, the set of arguments that a rational agent will accept is a set of arguments which can defend itself against all attacks on it.

Definition (Acceptable): An argument $A \in AR$ is acceptable with respect to a set S of arguments if and only if for each argument $B \in AR$: if $R(B, A)$ then there exists an argument in S that attacks B. [17]

Considering the argumentation framework in Figure 5, argument A is acceptable with respect to the set of arguments S = $\{A, C\}$ because although A is attacked by B, B is itself attacked by C which is in S. This is the same as saying that C defends A against B.

Intuitively, if we are trying to determine a set of winning arguments then the arguments within this set shouldn't attack each other — the set should be *conflict-free*.

Definition (Conflict-free): A set S of arguments is said to be conflict-free if there are no arguments A and B in S such that $R(A, B)$. [17]

In Figure 5, the set of arguments $\{A, D, F\}$ is conflict-free, whilst the set $\{A, B, F\}$ is not conflict free as $R(B, A)$.

Combining these ideas, if we can find a set of arguments that is conflict-free and in which each argument is acceptable then we have a rationally coherent set of arguments that we can defend against any argument not contained in the set. Such a set of arguments is said to be *admissible*.

Definition (Admissible): A conflict-free set of arguments S is admissible iff each argument in S is acceptable with respect to S. [17]

Consider a scenario where we have to decide a course of action and we are given an initial claim in favour of a particular decision and a set of related arguments for and against it. We might hope that if we can represent these arguments using an AF and find the admissible set of arguments then we just need to see whether the initial claim is included in that set. If it's in the admissible set then we'll be in favour of taking that action, and not in favour of it otherwise. Unfortunately it typically isn't this straightforward because

a given AF can have multiple admissible sets, some including the initial claim and some not. One way to reduce the number of admissible sets being considered is to only consider sets known as *complete extensions*.

Definition (Complete Extension): An admissible set S of arguments is called a complete extension iff each argument, which is acceptable with respect to S, belongs to S. [17]

Dung described the complete extension as representing *"the kind of confident rational agent who believes in every thing he can defend"*. Whilst restricting ourselves to complete extensions makes some useful progress in cutting down the number of interesting admissible sets, it is still possible for argumentation frameworks to have multiple complete extensions. For example, Figure 6 sets out the three possible complete extensions associated with the AF from Figure 5. The multiple extensions arise because arguments E and F attack each other, creating a cycle in the graph.

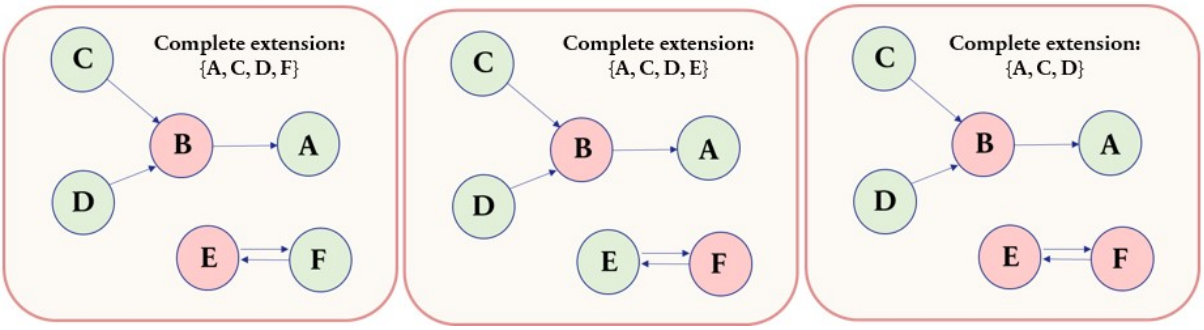


Figure 6: *The complete extensions of the argumentation framework in Figure 5.*

At this point, we could further subdivide the set of complete extensions into other more specific types of extensions (see Caminada [12]) and assess whether we could determine a method from these to construct a single 'best' set of acceptable arguments. Alternatively, if we limit ourselves to considering only finite, acyclic argument frameworks then it has been shown that there will only exist a single complete extension [17]. Whilst this does impose some limits on the type of arguments that can be represented, I have chosen to apply this restriction to my system as it will avoid the issues associated with analysing multiple complete extensions whilst still providing sufficient expressivity to represent a vast range of arguments.

Whilst this means it is not possible for my system to model two arguments that attack each other, the way in which the systems use the subjective user input to evaluate an argument framework actually results in a similar overall outcome in many instances, particularly for the preference-based and voting-based systems. More complex arguments involving large cycles also tend to be relatively rare. Furthermore, several notable extensions to Dung's framework that I have considered for use in my system (including the weighted bipolar framework which is used in my voting-based system) require argumentation frameworks to be acyclic to enable them to be evaluated and guarantee that certain desirable properties hold [3] [42].

Figure 7 gives examples of cyclic frameworks that my system won't be able to model. Specifically, in the middle framework an argument cycle exists (A, E, D, B) and in the right handside framework a cycle exists because two arguments attack each other (B, C).

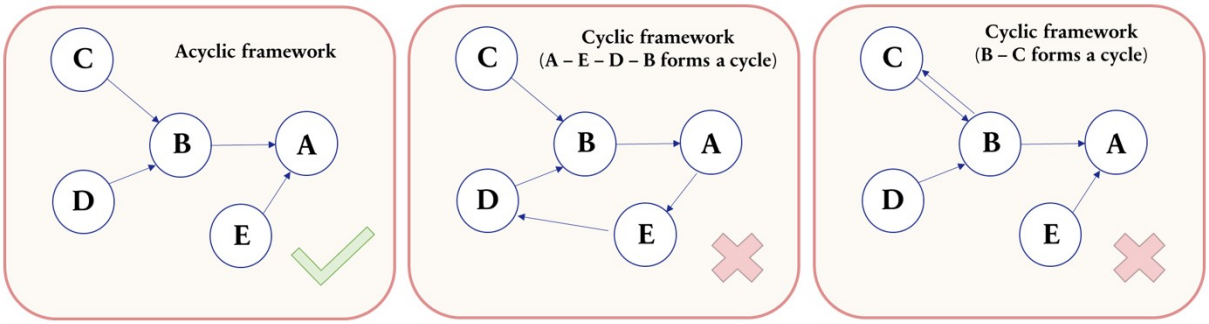


Figure 7: *Examples of acyclic and cyclic frameworks.*

4.2 Bipolar Argumentation Frameworks

So far we are limited to only modelling arguments that attack other arguments. I now introduce supporting arguments into this framework, moving from a unipolar to a bipolar model. For my systems I used the following definition of an Bipolar Argumentation Framework from [5]:

Definition (Bipolar Argumentation Framework): A bipolar argumentation framework (BAF) is a triple $<AR, R_{att}, R_{sup}>$, where AR is a set of arguments, R_{att} is a binary relation on AR called an attack relation and R_{sup} is another binary relation on AR called a support relation: consider A and $B \in AR$, $R_{att}(A,B)$ (resp. $R_{sup}(A, B)$ means that A attacks B (resp. A supports B).

Whilst the addition of the new support relation appears relatively straightforward, evaluating the framework to decide which arguments to accept becomes significantly more difficult. Imagine a simple framework consisting of an initial claim which has one argument supporting it and another attacking it. How do we decide if we should accept the initial claim or not? Alternatively, if we add another supporting argument does that mean we should now accept the initial claim given the number of supporters now outnumbers the attackers? If so, the system risks prioritising quantity of arguments over quality of arguments. Someone could game the system by knowing that they could overcome any other argument by putting forward a sufficiently large enough number of counterarguments regardless of how coherent or convincing they are. The method used to evaluate the framework and tackle these issues will depend on which of the three methodologies is being used.

In the following subsections I set out the methodology for the three extensions of the BAF: the preference-based system, the value-based system, and the voting-based system.

4.3 System 1: Preference-Based System

For the Preference-Based System the key implementation steps are as follows:

1. A BAF is created from the given set of arguments. The initial claim represents the proposed decision.
2. Each user is asked independently to provide their preferences between the arguments, except for the initial claim. These preferences are combined with the BAF to create a Preference-Based Bipolar Argumentation Framework (Pref-BAF) for each user.
3. Each user's Pref-BAF is evaluated (using their respective individual preferences) to determine their set of acceptable arguments.
4. The user frameworks are aggregated to determine the overall strength of each argument, for example if an argument is acceptable in the majority of individual frameworks then the system will be in favour of it.
5. Specifically, if the initial claim is acceptable in the majority of individual user frameworks then the system will be in favour of it.

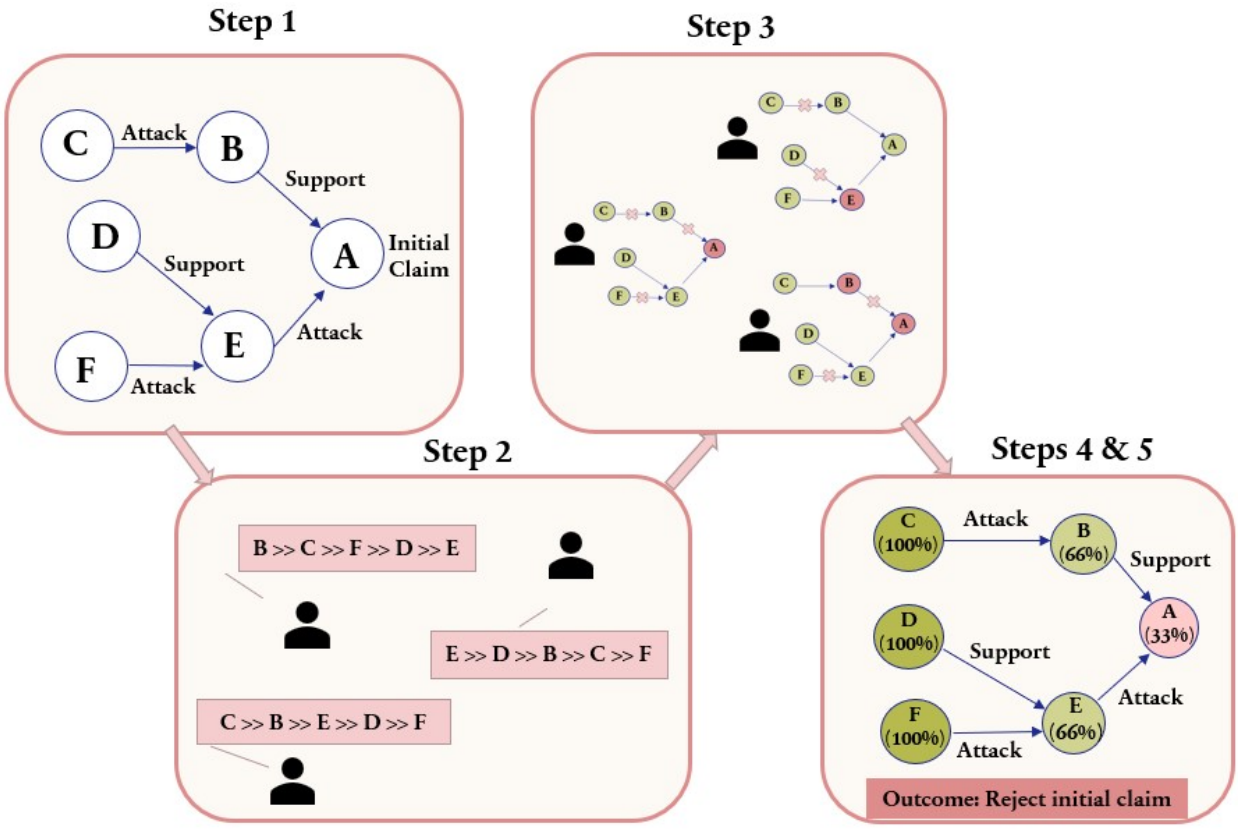


Figure 8: *The implementation steps for the Preference-Based System.*

4.3.1 Step 2: Creating the Preference-Based Argumentation Framework

For my preference-based system, users will be required to provide a preference ordering between the arguments, which will be used to evaluate the BAF. To do this I have expanded upon the Preference-Based Argumentation Frameworks in [4]:

Definition (Preference-Based Bipolar Argumentation Framework): A preference-based bipolar argumentation framework (Pref-BAF) is a quadruple Pref-BAF = $<AR, R_{att}, R_{sup}, Pref>$ where AR is a set of arguments, R_{att} is a binary relation on AR called an attack relation, R_{sup} is another binary relation on AR called a support relation, and $Pref$ is a preordering on $AR \times AR$. A preference for A over B is denoted as $A >> B$.

I have decided that whether the initial claim is accepted or not should be entirely determined by the subsequent arguments. To implement this in my system I set the initial claim to be the least preferred of all arguments. A consequence of this is that the initial claim will be defeated if it is attacked by at least one argument and has no supporters.

4.3.2 Step 3: Evaluating the Preference-Based Argumentation Framework

Other than the inclusion of supporting arguments, the key difference to Dung's original framework is how the preference ordering between the arguments impacts the evaluation of the framework. A revised notion of defeat (for a unipolar framework) is given in [4]:

Definition (Defeat in a Preference-Based Argumentation Framework): For $A, B \in AR$. B defeats A iff $R(B, A)$ and $B >> A$.

If we have argument A, which is attacked by argument B, then if $A >> B$ it can defend itself against the attack and so won't be defeated. Contrast this to Dung's original framework where argument A could only be defended by introducing another argument C that attacked B. We will now look at expanding this definition to consider the impact of supporting arguments and resolve the issues described at the end of Section 4.2.

Consider a simple framework where we have an argument A, which has one attacker B and one supporter C. Within this framework there are three scenarios:

1. $A >> B$
2. $C >> B >> A$
3. $B >> A$ and $B >> C$

For each scenario we need to determine whether A is an acceptable argument. In the first scenario, it's clear that we should accept A - if we ignore the supporting argument C then we would have a unipolar Preference-Based Argumentation Framework in which A would defend itself against B and so be acceptable. Adding C back into the framework will only strengthen A.

In the other two cases if we ignored C then in a unipolar Preference-Based Argumentation Framework A would be defeated by B. So does the presence of C mean we can defend A

against B in either of the two cases?

In the second scenario, as $C >> B$ then in some sense C is the better argument. Therefore, for my system I have decided that C would provide sufficient support to A to prevent it from being defeated by B.

For the third scenario I believe there are two reasonable approaches. The first is that because B is the most preferred argument it defeats A regardless of the support provided by C. The second approach considers whether the combined strength of A and C can overcome B. We might assign a score to each argument based on their preference ranking, with more preferred arguments being given a higher score. Then if the total score of A and its supporter C is higher than the score of the attacker B we could consider the attack to be unsuccessful.

Whilst I don't think there is optimal solution for all situations, I have chosen to take the former approach. When we consider scenarios with multiple attackers and supporters this approach avoids the 'quantity over quality' issue I discussed earlier, where a user might try to game the system by putting forward lots of weak arguments knowing that enough of them will eventually overcome any other argument no matter how strong they are. The downside to this approach is that there will be legitimate scenarios where a combination of weaker arguments should overcome a single stronger argument but my system will not be able to reflect this.

I therefore define what it means for argument A in a finite, acyclic Pref-BAF to be acceptable as follows:

Definition (Acceptable in a Pref-BAF): An argument $A \in AR$ is acceptable with respect to a set S of arguments if for each argument $B \in AR$: if $R_{att}(B, A)$, and $B >> A$ then at least one of the following holds:

1. There exists $C \in$ S such that $R_{att}(C, B)$, $C >> B$ and C is itself an acceptable argument.
2. There exists $D \in$ S such that $R_{sup}(D, A)$, $D >> B$ and D is itself an acceptable argument.

In Figure 9 this definition is applied to the three scenarios considered earlier.

Checking if these conditions hold will typically be a recursive process as it requires checking if the supporters and attackers of an argument are themselves acceptable. This could cause issues in frameworks containing cycles or with infinitely long argument chains. However, I have restricted the system to finite, acyclic frameworks which avoid these potential problems, which guarantees that the last argument in any argument chain will not have any attackers or supporters and so will be acceptable (in the graphical representation this corresponds to all arguments which are leaf nodes).

Furthermore, if we take S to be all the arguments in the argumentation framework then using this definition there is a unique set of acceptable arguments. Therefore, the system

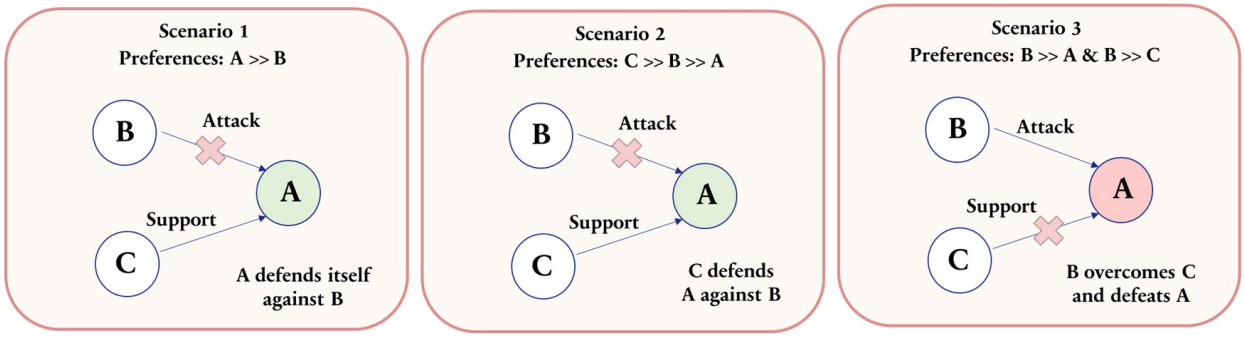


Figure 9: *A summary of the evaluation methodology for Pref-BAFs.*

only needs to check whether the initial claim is in this set and if so it will be in favour of it.

An example of evaluating a Pref-BAF is shown in Figure 10.

4.3.3 Steps 4 & 5: Aggregating Results of Multiple Users

Now we consider how to aggregate the results for multiple users. I considered two options for my system:

1. Evaluating the Pref-BAF for each individual user based on their own preferences. The system will be in favour of the initial claim if it is found to be acceptable in the majority of the individual user evaluations.

2. Combining the preferences of the users to create a single preference ordering, which can then be used to carry out a single evaluation of the Pref-BAF. The system will be in favour of the initial claim if it is found to be acceptable in this single evaluation.

As demonstrated by Figure 11, the chosen method can change the system outcome. In the example Pref-BAF each individual prefers a different single supporting argument over the sole attacker B. Therefore, when the framework is evaluated for each individual their preferred supporting argument will defeat B and A will be accepted. However, if the preferences are aggregated first by considering the most preferred argument in every pairwise comparison, B becomes the most preferred argument. Now when the framework is evaluated with these aggregated preferences B overcomes all the supporters and defeats A.

Of course there exist different methods of aggregating group preferences, and the issue of finding an optimal method has been studied extensively as part of social choice theory. Two important results from this field are Arrow's impossibility theorem [6] and the Gibbard-Satterthwaite theorem [23] [46], which broadly state that without applying any restrictions it is impossible to find a way to aggregate the user preferences in such a way that satisfy certain basic, desirable properties.

For example, Arrow's impossibility theorem means that when there are more than two

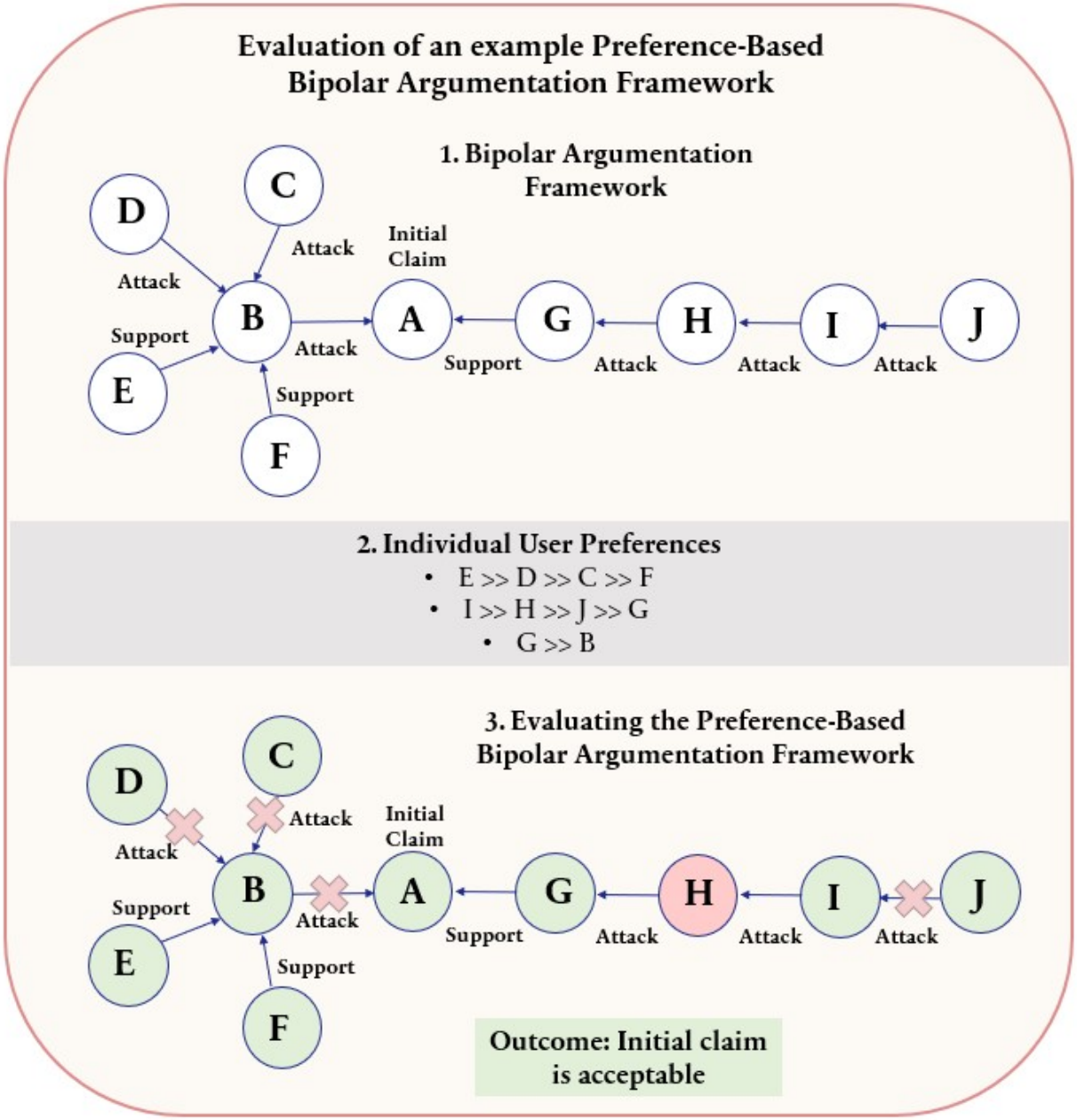


Figure 10: *An evaluation of an example Pref-BAF.*

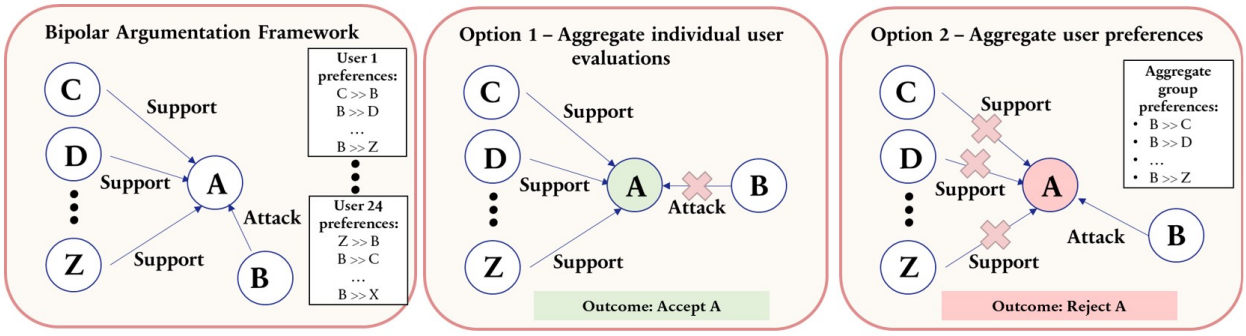


Figure 11: *A comparison of the two methods for aggregating user preferences in a Pref-BAF.*

arguments being considered, there does not exist a preference aggregation function that satisfies the following criteria (adapted from [33]):

- Weakly pareto: if all users prefer argument A over argument B, then the group should also prefer argument A over argument B.
- Independence of irrelevant alternatives: if every user's preference between argument A and argument B remains unchanged, then the group's preference between argument A and argument B should also remain unchanged, even if users' preferences about other pairs of alternative arguments change.
- Non-dictatorship: there is no single user whose preferences always determine the group's preferences.

We can recover these properties if we put in place certain restrictions, for example Black showed that for an odd number of users with single-peaked preferences (a type of restriction on possible preference orderings) we can get an ordering for which Arrow's theorem and the Gibbard-Satterthwaite theorem do not apply [10]. However, for many settings applying the required restrictions will be impractical and likely require additional input from either the users (which might be subject to manipulation) or a system moderator. Furthermore, the restrictions imposed on users may cause confusion or frustration as they would be limited as to what preference orderings they could choose.

Given these issues associated with aggregating preferences, I believe the most appropriate way to account for multiple users is to aggregate the individual user argumentation framework evaluations instead. This is done for each argument by calculating the proportion of user evaluations in which it is acceptable.

4.3.4 System Output

The system outputs a value for each argument representing the proportion of user evaluations in which that argument is acceptable. Specifically, if the initial claim is acceptable in the majority of user evaluations then the system is in favour of it.

Pseudocode for the evaluation of a Pref-BAF is given in Figure 12. For an implementation of this system using Python please see Appendix B.

Preference-Based System Evaluation Pseudocode

Input: Arguments, Attacks, Supports, User preference orderings
Output: Group acceptance values for each argument

```
For each argument:
    group_acceptance(argument) ← 0

For each user_preference_ordering:
    For each argument:
        If argument has no attackers:
            user_acceptance(argument) ← 1
        Else:
            If there exists an attacker, s.t. attacker >> argument,  attacker >> supporter for all supporters of argument with
                user_acceptance(supporter) = 1, and user_acceptance(attacker) = 1:
                    user_acceptance(argument) ← 0
            Else:
                    user_acceptance(argument) ← 1
        group_acceptance(argument) ← group_acceptance(argument) + user_acceptance(argument) / (number of users)
```

Figure 12: *Pseudocode for the evaluation of a Pref-BAF.*

4.4 System 2: Value-Based System

For the Value-Based System the key implementation steps are as follows:

1. A BAF is created from the given set of arguments. The initial claim represents the proposed decision.
2. Arguments promoting values are identified to create a Value-Based Bipolar Argumentation Framework (Val-BAF).
3. Each user is asked independently to provide their preferences between these values.
4. The Val-BAF is evaluated for each user using their respective value preferences to determine a probability of acceptance for each argument.
5. The user evaluations are aggregated to determine an average probability of each argument being accepted.
6. Specifically, if the initial claim has an average acceptance probability of over 0.5 then the system will be in favour of it.

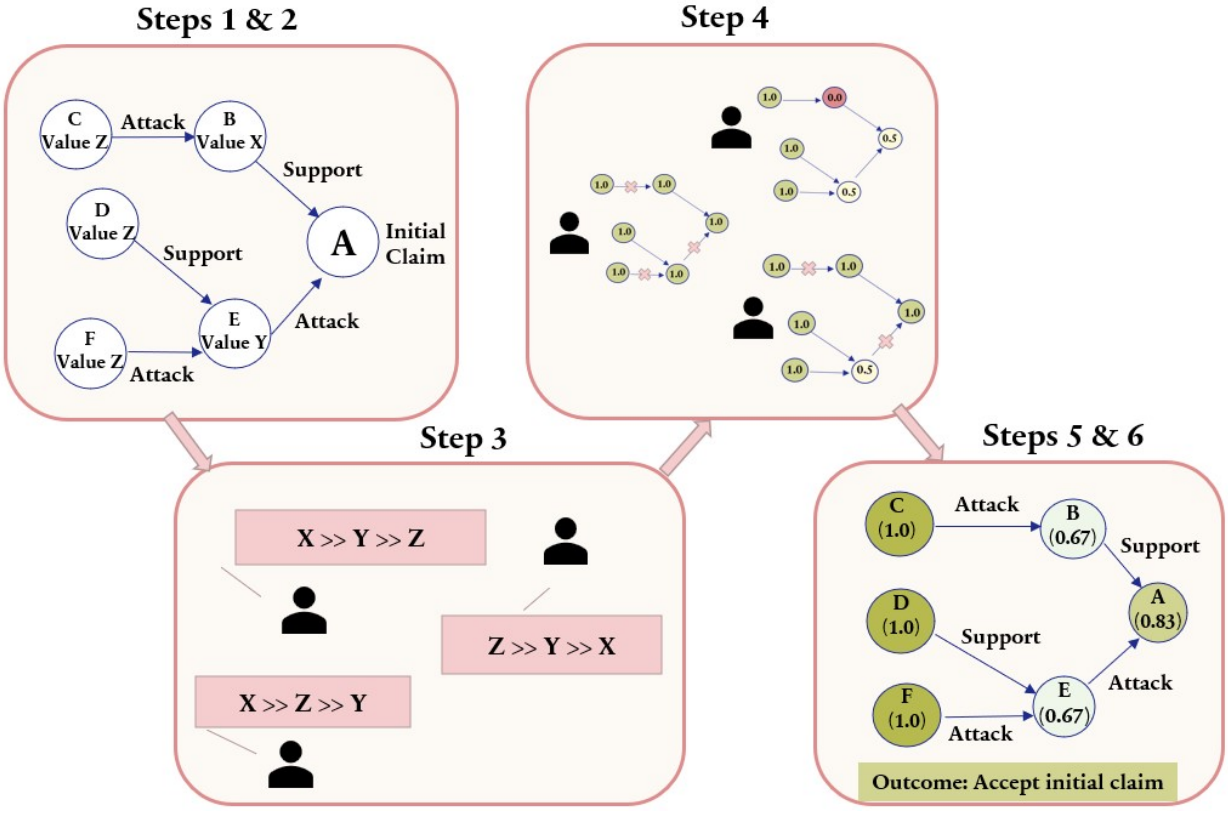


Figure 13: *The implementation steps for the Value-Based System.*

4.4.1 Steps 2 & 3: Creating the Value-Based Argumentation Framework

For my value-based system, values being promoted by the arguments in the underlying BAF will need to be identified (noting multiple arguments may promote the same value)

and users will be required to provide their preferences between these values, which will be used to evaluate the BAF. To allow for this I have expanded upon the Value-Based Argumentation Frameworks in [8]:

Definition (Value-Based Bipolar Argumentation Framework): A value-based bipolar argumentation framework (Val-BAF) is a 6-tuple Val-BAF = $<AR, R_{att}, R_{sup}, V, val, Valpref>$ where AR, R_{att}, R_{sup} are as before, V is a non-empty set of values, val is a function that maps arguments from AR to values in V. An argument A relates to a value v if accepting A promotes or defends v, the value in question is given by $val(A)$. For every $A \in AR$, $val(A) \in V$. $Valpref$ is a preordering on $V \times V$. We denote a preference for value v_1 over v_2 as $v_1 >> v_2$.

If every argument is related to a different value then this framework becomes equivalent to a Pref-BAF, whilst if all arguments are related to the same value then it is equivalent to a regular BAF. Similar to Pref-BAFs I have made an assumption for my system that the value associated with the initial claim is the least preferred of all values, so that the decision is determined entirely by the subsequent arguments.

4.4.2 Step 4: Evaluating the Value-Based Argumentation Framework

As with Pref-BAFs we need to be able to determine what arguments are acceptable in a Val-BAF. Again, we start with a revised notion of defeat (for a unipolar framework) as given in [8]:

Definition (Defeat in a Value-Based Argumentation Framework): For $A, B \in AR$. B defeats A if $R(B, A)$ and $val(A)$ is not preferred to $val(B)$.

This is similar to how defeat was defined for a Preference-Based Argumentation Framework, except that the success of an attack is now determined by looking at the preferences between the values associated with the arguments rather than the arguments themselves. However, an important difference is that several arguments can promote the same value. This can cause issues in a bipolar framework. For instance, if there is an argument with a single attacker and a single supporter, all promoting the same value, how can we determine if the initial argument is defeated or not? In the Pref-BAF I decided that the most preferred argument would be used to determine the outcome, but that is no longer possible.

Instead I have created a new, more probabilistic approach to evaluate the framework, where if there is a set of arguments that all appear equally strong then I assume there is an equal probability that each of them is the strongest. Therefore, unlike for Pref-BAFs, each argument now has a probability between 0 and 1 of being accepted. In order to define this more formally:

- Let $att(A)$, $sup(A)$ and $pre(A)$ denote the sets containing the attackers of A, the supporters of A, and the union of these two sets, respectively.
- Let $pre^v(A) = \{B \in AR | val(B) = v, B \in pre(A)\}$.

- Let $pre^{v+}(A) = \{B \in AR | val(B) >> v, B \in pre(A)\}$.
- Let 2^Y denote the powerset of set Y.

Then for each argument in a Val-BAF its acceptance probability is defined as follows:

Definition (Acceptance Probability in a Val-BAF): The acceptance probability for an argument in a Val-BAF is a function defined as $f: AR \mapsto [0,1]$ where, for any $A \in AR$:

$$p_{acc}(A) = 1 - \sum_{\substack{v \in V, \\ s.t.\ v \gg val(A) \\ or\ v = val(A)}} \sum_{\omega \in 2^{pre^v(A)}} \left(\prod_{a \in \omega} (p_{acc}(a)) \cdot \prod_{b \in pre^v(A)/\omega} (1 - p_{acc}(b)) \cdot \prod_{c \in pre^{v+}(A)} (1 - p_{acc}(c)) \cdot \frac{|att(A) \cap \omega|}{|\omega|} \right)$$

where $|\omega|$ and $|att(A) \cap \omega|$ are the total number of arguments in ω and in the intersection of ω and $att(A)$ respectively.

For each value which is the same as or more preferred than the value of A, the function considers possible combinations of arguments promoting that value being acceptable or not. A specific combination of acceptable arguments promoting a given value is represented by ω, arguments in this set are accepted and arguments promoting the given value that are not in the set are not accepted. The formula in the definition calculates the probability of this combination occurring - the first product calculates the probability that the arguments in ω are accepted and the second product calculate the probability that the arguments not in ω are not accepted. The third product then calculates the probability that all the arguments promoting more preferred values are not accepted, for if at least one of them was accepted then the arguments promoting less preferred values would become irrelevant.

The final term on the right then calculates the proportion of attacking arguments in ω to determine the probability that the attack would be successful if the specific combination of arguments in ω occured. It assumes that for arguments promoting the same value they all have an equal chance of being the strongest argument.

Typically, this will be a recursive calculation but given the system's restriction to finite, non-cyclic frameworks it will always be well-defined. It also assumes that the acceptance probabilities of the attackers and supporters are independent of each other, which may not always be the case in complex frameworks. Some worked examples of this formula are provided in Appendix A.

Unfortunately, this approach is susceptible to the 'quantity over quality' problem, as an argument is more likely to be defeated the more attackers (promoting the same or a more preferred value) it has.

An example of evaluating a Val-BAF for an individual is shown in Figure 14.

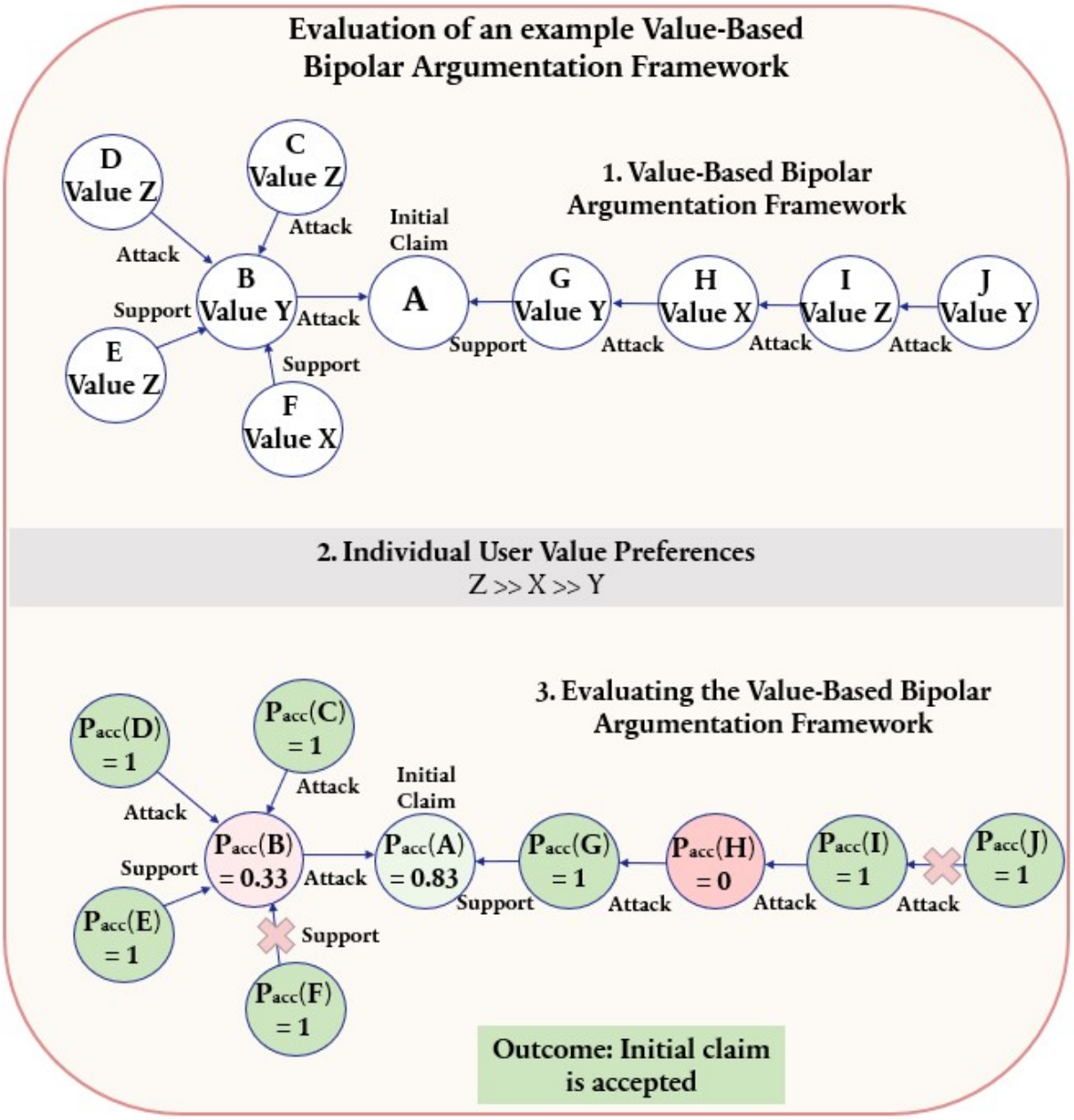


Figure 14: *An evaluation of an example Val-BAF.*

4.4.3 Steps 5 & 6: Aggregating Results of Multiple Users

As with Pref-BAFs, frameworks for multiple users could be evaluated by either aggregating the results of individual evaluations or by aggregating the user value preferences to evaluate a single framework. For the same reasons as the Pref-BAFs I have decided to take the former approach. The method is similar to the Pref-BAFs, except that rather than calculate the proportion of users that find a given argument to be acceptable the average acceptance probability for each argument is now calculated.

4.4.4 System Output

The system outputs a value for each argument representing the average acceptance probability amongst all the user evaluations. Specifically, if the initial claim has an average acceptance probability higher than 0.5 then the system is in favour of it.

Pseudocode for the evaluation of a Pref-BAF is given in Figure 15. For an implementation of this system using Python please see Appendix C.

Value-Based System Evaluation Pseudocode

Input: Arguments, Values, Attacks, Supports, User value preference orderings
Output: Group acceptance probabilities for each argument

```
For each argument:
    group_acceptance_prob(argument) ← 0

For each user_value_preference_ordering:
    For each argument:
        If argument has no attackers:
            user_acceptance_prob(argument) ← 1
        Else:
            preferred_values = {set of values that are the same as or more preferred than val(argument)}
            successful_attack_prob ← 0
            For each preferred_value:
                preferred_value_arguments = {set of arguments for which preferred_value is the associated value and
                it is either an attacker or supporter of argument}
                more_preferred_value_arguments = {set of arguments for which its associated value is preferred to
                preferred_value and it is either an attacker or supporter of argument}
                For each ω in the powerset of preferred_value_arguments:
                    If ω is not empty:
                        accepted_arguments_prob_ω ← ∏ (user_acceptance_prob(args)), for args in ω
                        not_accepted_arguments_prob_ω ← ∏ (1 - user_acceptance_prob(args)), for args in
                                {preferred_value_arguments \ ω} ∪ {more_preferred_value_arguments}
                        proportion_of_attackers_ω ← proportion of arguments in ω that attack argument
                        successful_attack_prob_ω ← accepted_arguments_prob_ω × not_accepted_arguments_prob_ω ×
                                proportion_of_attackers_ω
                        successful_attack_prob ← successful_attack_prob + successful_attack_prob_ω
            user_acceptance_prob(argument) ← 1 - successful_attack_prob
        group_acceptance_prob(argument) ← group_acceptance_prob(argument) + user_acceptance_prob(argument) /
                (number of users)
```

Figure 15: *Pseudocode for the evaluation of a Val-BAF.*

4.5 System 3: Voting-Based System

For the Voting-Based System the key implementation steps are as follows:

1. A BAF is created from the given set of arguments. The initial claim represents the proposed decision.
2. Each user is asked independently to vote for each argument using a scale from 1-6 (1 representing a weak argument, 6 representing a strong argument), except for the initial claim.
3. The individual votes are then aggregated using a voting aggregation function to determine the vote base scores for each argument. The initial claim is assigned a vote base score of 0.5.
4. The resulting weighted argumentation framework is then evaluated using exponent-based restricted semantics to determine the final weights of each argument.
5. If the final weight of the initial claim is above 0.5 then the system will be in favour of it.

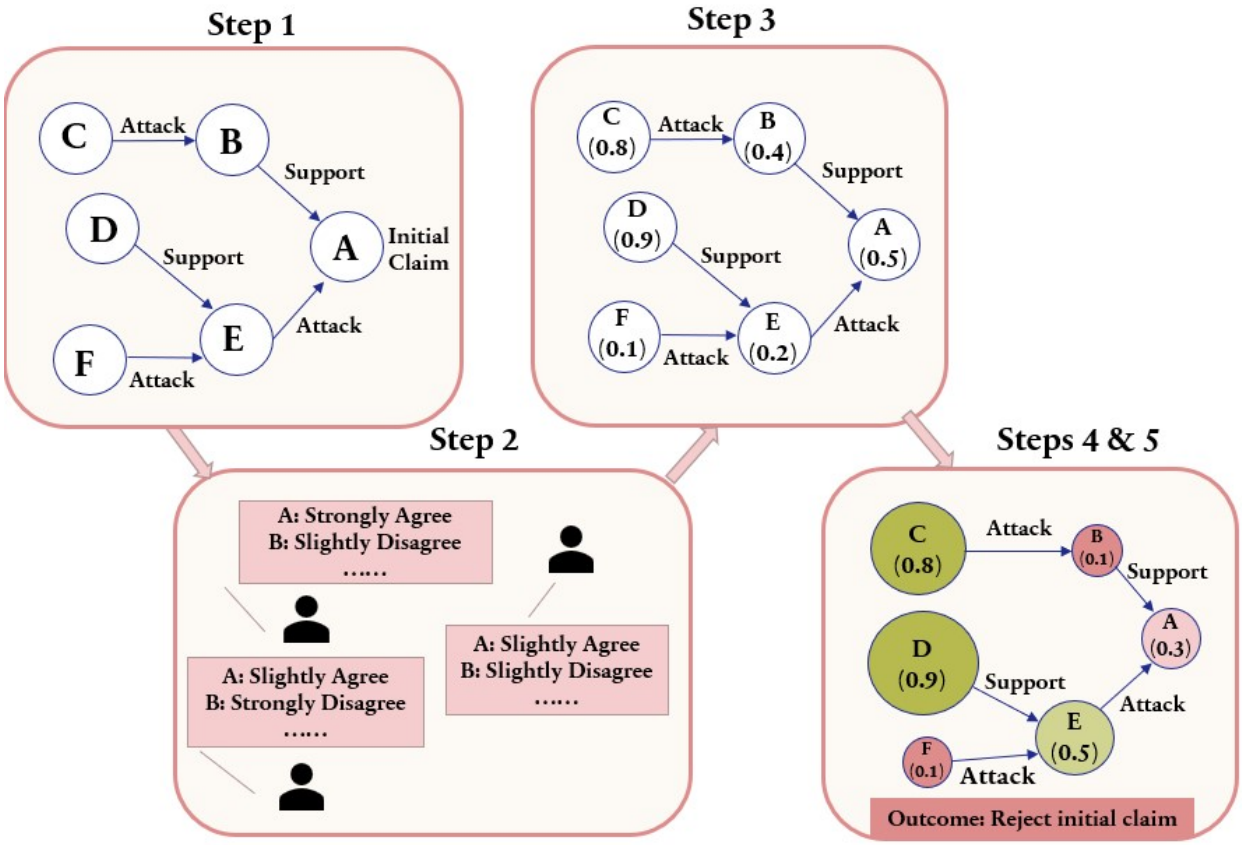


Figure 16: *The implementation steps for the Voting-Based System.*

4.5.1 Step 2: Creating the Voting-Based Argumentation Framework

For my voting-based system, users are required to submit a weighted vote for each argument which is used to evaluate the BAF. To do this I have expanded upon the Social Abstract Argumentation Framework in [28] (which only allows votes for or against an argument rather than weighted votes):

Definition (Voting-Based Bipolar Argumentation Framework): A voting-based bipolar argumentation framework (Vote-BAF) is a 6-tuple Vote-BAF = $<AR, R_{att}, R_{sup}, X, W, V>$ where AR, R_{att}, R_{sup} are as before, X is a set of voting options (eg {for, against}), $S{:}X \mapsto [0,1]$ is a function that assigns a score between 0 and 1 to each voting option, and $V{:}AR \times X \mapsto \mathbf{N}$ is a function that counts the number of votes for a given argument and voting option.

Whilst the voting framework in [28] only gives users a binary choice to either vote for or against an argument, the above definition allows for more flexibility in possible voting options. In my system I have set my voting options $X = \{1, 2, 3, 4, 5, 6\}$, in which 1 represents a very weak argument and 6 a very strong argument. This is still a concise set of options, which although more expressive than a binary choice should not be too onerous for users. I have also avoided a neutral option as this forces users to take a position on each argument. There are existing frameworks that do include such a neutral option, for example Rago & Toni's QuAD-V Framework [41]. In line with the approach taken in the preference and value systems, I have decided that users won't vote on the initial claim.

For my vote options, I have defined S as:

$$S(x) = \begin{cases} 0 & \text{if } x = 1, \\ 0.2 & \text{if } x = 2, \\ 0.4 & \text{if } x = 3, \\ 0.6 & \text{if } x = 4, \\ 0.8 & \text{if } x = 5, \\ 1 & \text{if } x = 6. \end{cases}$$

In practice, the actual weights for the function can be adjusted following analysis of the system's performance. However, I believe an equal partition of the unit interval is a sensible starting point.

4.5.2 Step 3: Aggregating Votes of Multiple Users

The next step is to define a function that outputs a single base score for each argument using its corresponding votes. The below is a generalisation of similar functions from [41] and [28].

Definition (Vote Base Score): A vote base score is a function defined as τ:$AR \mapsto [0,1)$ where, for any $A \in AR$, and $\epsilon > 0$:

$$\tau(A) = \begin{cases} 0.5 & \text{if } |V(A)| = 0, \\ \frac{\sum_{x \in X} S(x)V(A,x)}{|V(A)|+\epsilon} & \text{otherwise.} \end{cases}$$

Where |V(A)| is the total number of all votes associated with A.

In my system, a vote base score close to 0 occurs when most users vote '1', and a base score close to 1 occurs when most users vote '6'. This function also assigns the initial claim a base score of 0.5. The inclusion of ϵ ensures the score is always less than 1, this is a requirement for applying the exponent-based restricted semantics in the evaluation step. I have set ϵ to be 0.01.

The vote base score of an argument A is denoted as $vbs(A)$.

4.5.3 Steps 4 & 5: Evaluating the Voting-Based Argumentation Framework

Next we evaluate the Vote-BAF by applying exponent-based restricted semantics, using the following function from [3]:

Definition (Exponent-Based Restricted Semantics): The restricted semantics is a function defined as f:$AR \mapsto [0,1)$ where, for any $A \in AR$:

$$f(A) = 1 - \frac{1 - vbs(A)^2}{1 + vbs(A)2^E} \text{ where } E = \sum_{x \in sup(A)} f(x) - \sum_{x \in att(A)} f(x)$$

Where $sup(A)$ is the set of arguments that support A and $att(A)$ is the set of arguments that attack A.

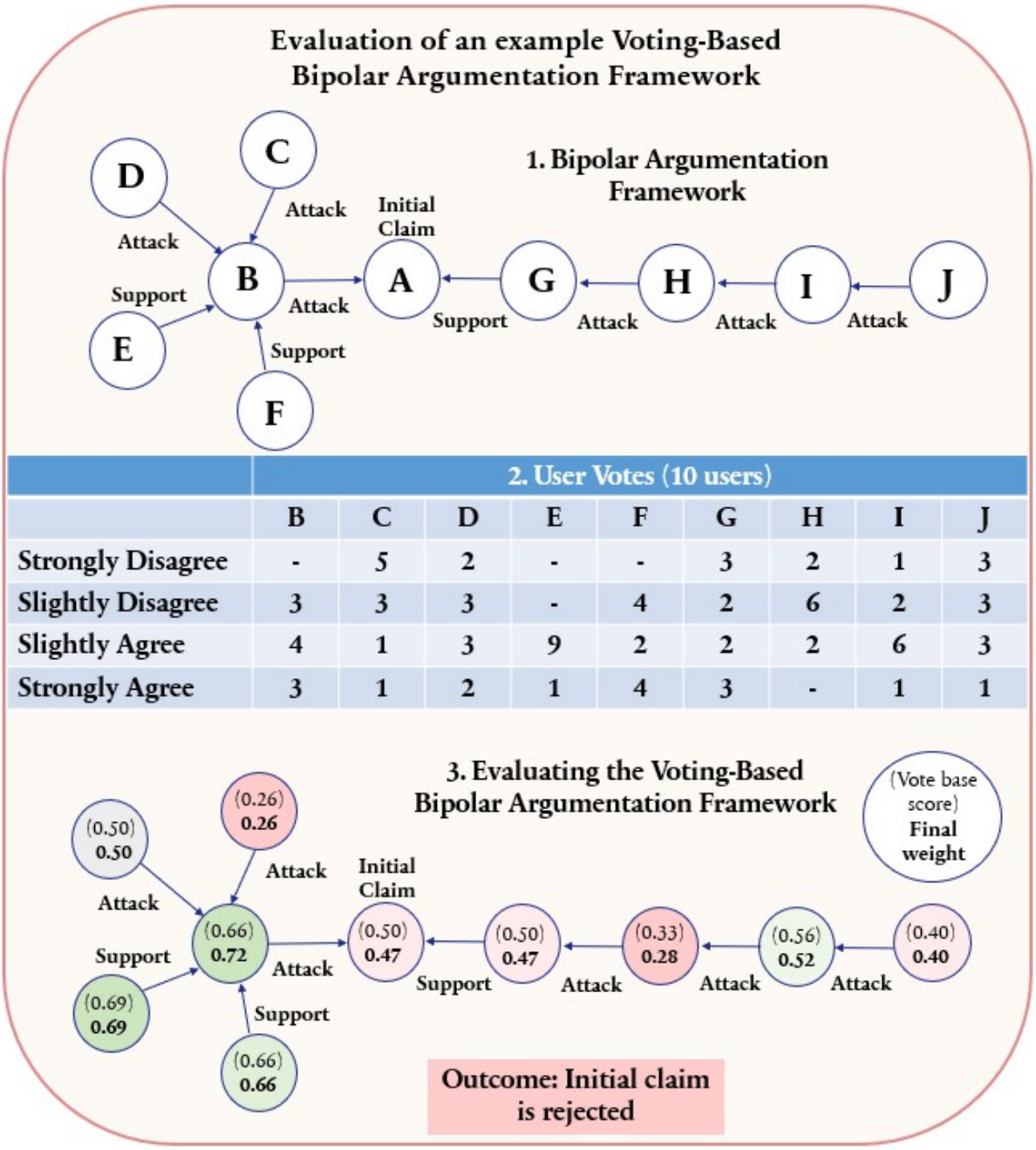
As expected, the final weight of an argument is higher the more supporters it has, the stronger those supporters are and the higher the initial vote base score. Meanwhile, the final weight for an argument will be lower the more attackers it has, the stronger those attackers are and the lower the initial vote base score.

The system will be in favour of the initial claim if its final weight is higher than 0.5 (its initial vote base score). Moreover, the higher the final weight the greater the level of acceptance of the initial claim. The system will also calculate final weights for every argument so it is possible to also evaluate their level of acceptance and the influence they have had on the final outcome.

An example of evaluating a Vote-BAF for a group is shown in Figure 17.

4.5.4 System Output

The system outputs a value for each argument representing its final weight. Specifically, if the initial claim has a weight higher than its vote base score (set as 0.5 in my system)



	B	C	D	E	F	G	H	I	J
Strongly Disagree	-	5	2	-	-	3	2	1	3
Slightly Disagree	3	3	3	-	4	2	6	2	3
Slightly Agree	4	1	3	9	2	2	2	6	3
Strongly Agree	3	1	2	1	4	3	-	1	1

Figure 17: *An evaluation of an example Vote-BAF.*

then the system is in favour of it.

Pseudocode for the evaluation of a Vote-BAF is given in Figure 12. For an implementation of this system using Python please see Appendix D.

Voting-Based System Evaluation Pseudocode

Input: Arguments, Attacks, Supports, User votes, Vote scores
Output: Final weights for each argument

For each *argument*:

- **If** *argument* has no user votes:
  - vote_base_score(*argument*) ← 0.5
- **Else:**
  - vote_base_score(*argument*) ← ($\sum number\ of\ votes\ \times vote\ score$) / (*number of users* + ε)

For each *argument*:

- **If** *argument* has no attackers or supporters:
  - final_weight(*argument*) ← vote_base_score(*argument*)
- **Else:**
  - E = $\sum$ final_weight($supporters$) - $\sum$ final_weight($attackers$)
  - final_weight(*argument*) ← $1\ -$ (1 $-$ vote_base_score(*argument*)^2) / (1 + vote_base_score(*argument*)2^E)

Figure 18: *Pseudocode for the evaluation of a Vote-BAF.*

5 Analysis & Evaluation

In this section I first evaluate each of the three systems against the system specifications from Section 3. After I provide details on the user testing I carried out, including the methodology used, its limitations and analysis of the results.

5.1 Review of the Systems Against the Specifications

1. *The system is required to model arguments and the relationships between them using a representation that the system can then analyse to output a decision based on those arguments.*

   Evaluation: Each system uses a BAF as the underlying model for representing arguments, and can be evaluated using the methodologies described in Section 4.

2. *Any modelling restrictions should not result in significant limitations on the situations in which the system can be applied.*

   Evaluation: As the systems do not require a semantic understanding of the actual arguments, and the abstract nature of the representations, they can be applied to a broad range of scenarios. However, each system is restricted to evaluating acyclic frameworks, although even with this restriction I believe the systems are all sufficiently versatile.

3. *The system is required to output a single decision based on the arguments and user input given. In any cases where a decision can't be made it should be possible to understand why.*

   Evaluation: Each system either outputs a single decision or is undecided as determined by the methodologies set out Section 4. In cases where the system is undecided (where the final value of the initial claim is 0.5) the additional system output and individual evaluations where available can be analysed further.

4. *The system is required to take into account pre-determined forms of subjective input from each of the users when making a decision. However, any inputs required from the users should be simple in nature and not too onerous for the users to provide.*

   Evaluation: Each system uses a different form of simple subjective user output to make its decision, although the size of the framework will impact how onerous it is to accurately provide this input.

   For the value-based system, typically multiple arguments will promote the same value, and so as the number of arguments grows the number of different values remains relatively unchanged. On the other hand, for both the preference-based and voting-based systems the amount of user input required increases in line with the size of the framework, which could become time-consuming to provide in large

frameworks.

However, a downside of the value-based system is that it may be difficult to initially assign appropriate values to the arguments. It is not always obvious which values are being promoted by arguments, how broad in scope the values should be, and how to handle arguments that seem to promote multiple values. This means there can exist different value labellings, which can impact the final outcome.

5. *The system is required to take into account the input from all of the users when making a decision. The system should not place greater weight on the input from some users over others.*

   Evaluation: The decision made by each system does take into account the input from all the users.

   For the preference-based system, the decision is determined by the proportion of user evaluations that consider the initial claim to be acceptable. As such, all users have an equal influence on the output of the system.

   For the value-based system, the decision is determined by the average acceptance probability of the initial claim amongst the users. This means that each user has the same ability to influence the outcome, although in practice the outcome will be influenced more by users with acceptance probabilities closer to 0 or 1. However, given the system methodology it will be difficult for the users to know how to engineer their input to achieve these extreme scores.

   For the voting-based system, each user has the same number of votes and so the same ability to influence the outcome. However, when a range of scoring options is available, users who provide more extreme voting choices will have a greater influence on the final outcome than users providing more moderate choices.

6. *The decisions made by the system should be explainable. This means it should be possible for the users to understand which arguments or factors have influenced the final decision, and ideally to what extent they have influenced that decision. The system should indicate its level of conviction in its decision.*

   Evaluation: The preference-based system calculates the proportion of user evaluations in which an argument is acceptable. The level of conviction in the decision can be determined by the proportion of user evaluations in which the initial claim is acceptable or not. It is also possible to calculate the proportion of times that a given attacking or supporting argument was successful. Influential arguments are those which are acceptable and successfully attack or support in the majority of user evaluations.

   The value-based system also provides similar metrics, but in this case based an

average probability of acceptance. However, one drawback in both these frameworks is that arguments that are not attacked will always be acceptable. If users are unaware of this fact, they might believe these arguments are stronger than they actually are.

Therefore, I believe the voting-system output is the simplest to understand as it calculates a single weight for each argument, determined by the votes its received and the strength of its attackers and supporters. In particular, even an argument with no attackers can have a low weight. Influential arguments are those with weights closer to 1. Similarly, the higher the final weight of the initial claim, the greater the conviction in the decision.

7. *The system should be designed in such a way to minimise the risk of individual users from being able to 'game' it.*

   Evaluation: The ways in which users can try to game the outcome of the system are via the:

   i) *Arguments put forward:* In all cases, users may choose to withhold important arguments if they weaken the case for their desired outcome. However, this is an issue for any user led argumentation system. The opposite issue is users putting forward arguments supporting their desired outcome regardless of their relevance - the 'quantity over quality' issue. The preference-based system was designed to defend against this behaviour by focusing on the most preferred arguments rather than the number of arguments. The voting-based system can defend against this issue if users vote down the less relevant arguments. However, the value-based is susceptible as putting forward a larger number of attacking arguments promoting the same value as the target argument will increase the probability of the target being defeated. Such cases rely on users putting forward counter-arguments to the less relevant arguments. See Figure 19 for an example.

   ii) *Relationships between arguments (supports and attacks):* Assuming users are responsible for assigning relationships between arguments, then some may do this in such a way to game the outcome. The extent to which this is an issue varies depending on the specific characteristics of the framework being considered. It is also more likely to be an issue for smaller frameworks, as for larger and more complex frameworks it will be harder for users to predict how changes to its structure will impact the final outcome. However, it is generally true that arguments directly connected to the initial claim will have greater influence on the outcome than those at the end of a long chain of arguments. As such, some users may state that a particular argument relates to the initial claim when really it relates to a different argument. As I have avoided asking users to comment on meta-structures, such as attacks on attacks, all the systems are exposed to this type of behaviour. That being said, if there are disagreements over the correct structure for the framework it would be possible

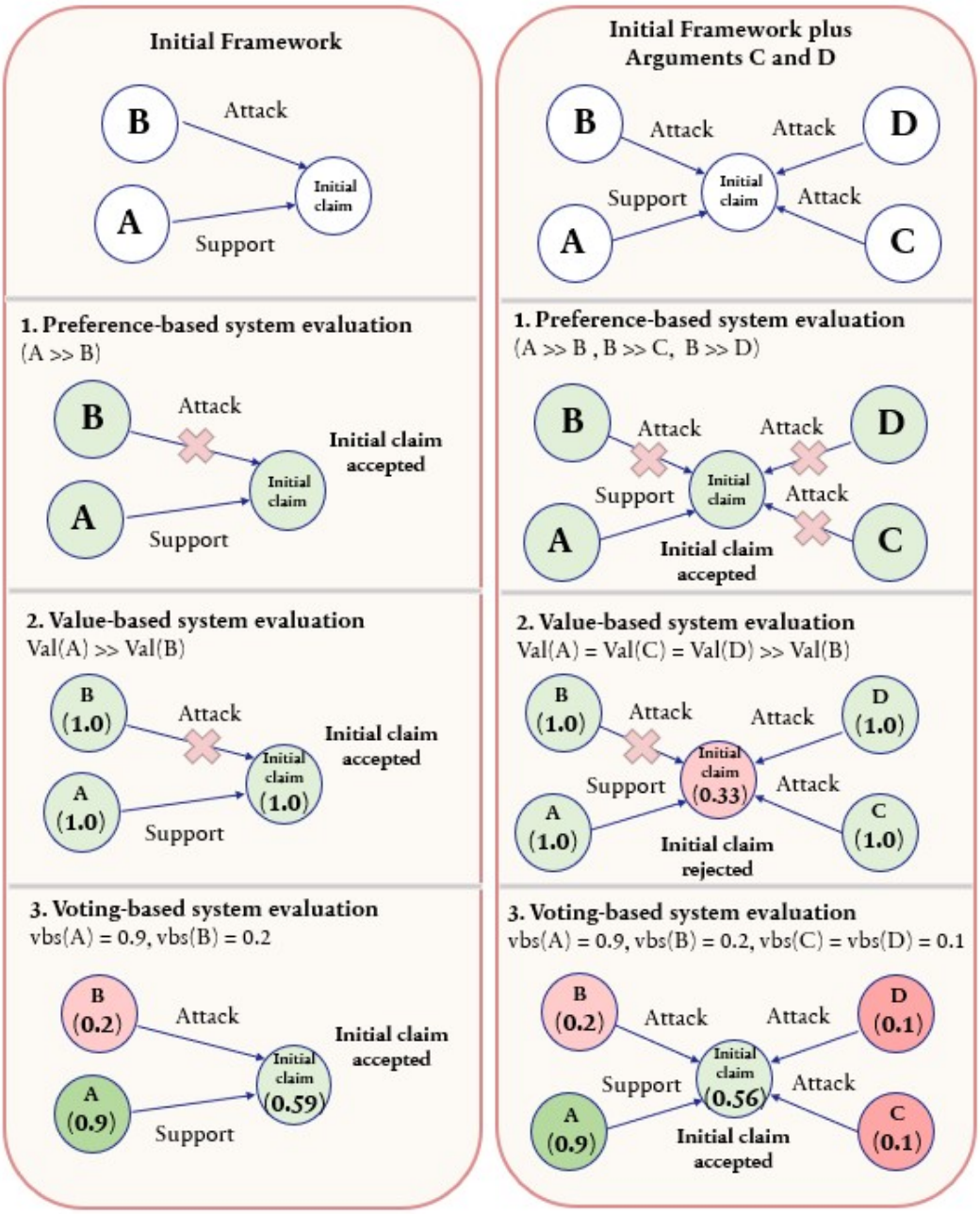


Figure 19: *An example of adding additional (weak) arguments under each system.*

to evaluate the different structures and compare differences in the outcome.

iii) *Values assigned to arguments:* Relevant to only the value-based system, users may try to link their arguments to more favoured social values, no matter how tenuous the link, so that they are more likely to be successful when evaluating the framework. A solution could be to appoint an independent moderator to assign values to the arguments, otherwise the only defence is to rely on other users putting forward counter-arguments.

iv) *Subjective user input:* Users may try to be tactical with how they provide their input. For the preference-based and value-based system, aggregating the final user

evaluations rather than the user input removes the incentive to misreport preferences. However, the voting-based system is susceptible to tactical voting, as users desiring a particular a outcome can increase the likelihood of that outcome by voting in favour of every argument supporting that outcome and against every other argument. Without additional restrictions, such as potentially standardizing each users' votes, there is no obvious way to defend against this other than by analysing individual user votes. See Figure 20 for an example.

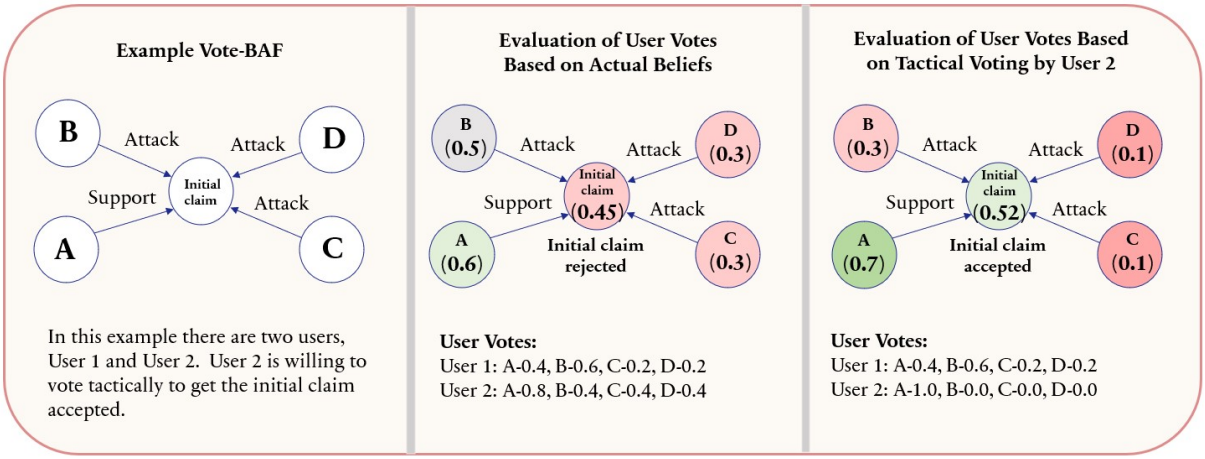


Figure 20: *An example of tactical voting in a Vote-BAF.*

5.2 User Testing Methodology

To test the three systems with actual users, I designed a hypothetical scenario requiring a decision to be made, constructed several arguments for and against the decision, and identified the relevant values associated with these arguments. In each test, a set of participants was identified and provided with the details of the hypothetical scenario and the arguments. Initially, before reading the scenario and arguments, each participant was independently asked to rank the importance of the values promoted by the arguments (the input to the value-based system). Then after reading the scenario and arguments the participants were independently asked to score each argument between 1-6 based on its perceived strength (the input to to the voting-based system) and provide a preference ordering between the arguments (the input to the preference-based system).

The details of the hypothetical scenario, arguments, user questionnaire and the scenario argumentation framework given to the participants is included in Appendix E. The underlying framework is shown in Figure 21.

Meanwhile, the relevant answers to the questionnaires were input into each of the three systems, which used this input to produce a group decision (one for each system). The value-based and preference-based systems were also able to produce a decision for each participant. The participants were then asked to discuss the scenario as a group to try to agree to a collective decision. Three participants were then chosen to independently

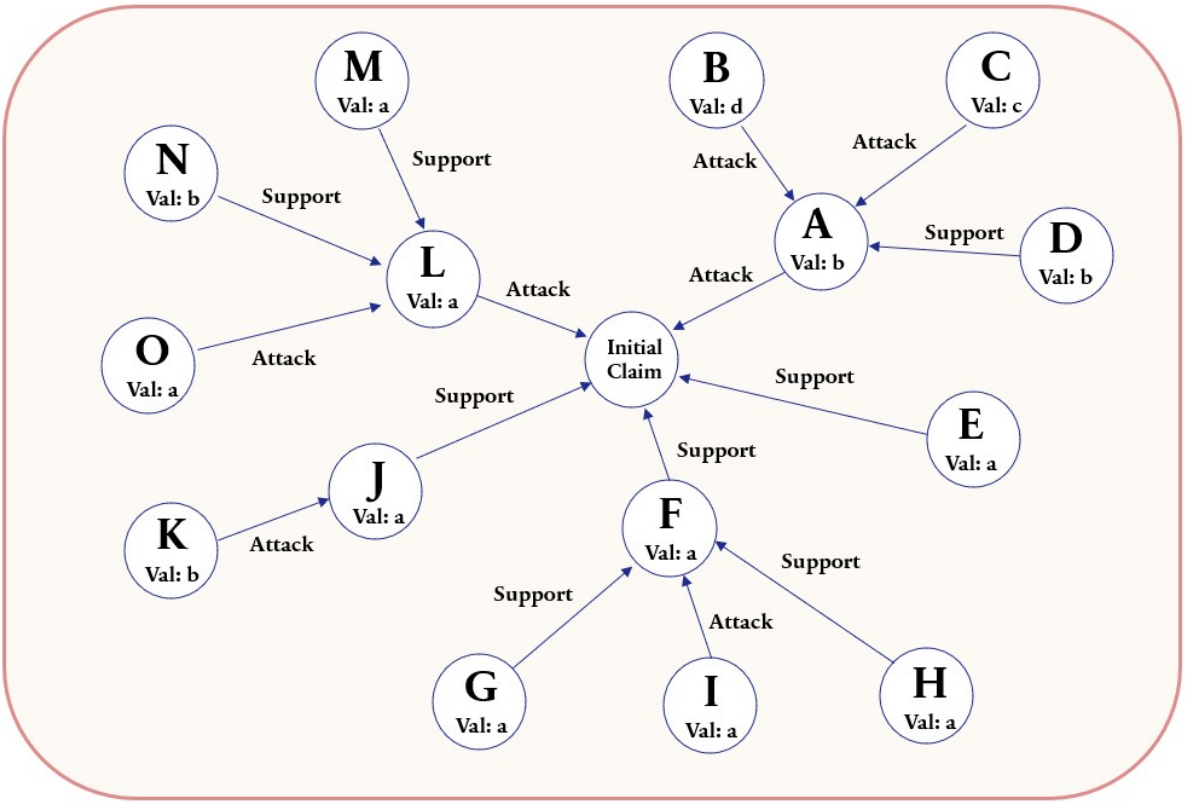


Figure 21: *The underlying Val-BAF used in user testing.*

provide feedback on the group output from one of the three systems (ie one participant per system).

5.3 Testing Limitations

Before discussing my results it is important to be aware of the limitations of this testing:

1. The number of results obtained is small relative to typical scientific studies and only considers one example scenario. This means my results may reflect idiosyncrasies specific to my set of participants or the scenario tested.

2. Given the small number of participants I did not test each of the systems independently. Whilst this allowed for more comparison between the systems it is possible that the testing of one system may have influenced the testing of the others.

3. My analysis is based on results using a hypothetical scenario. However, studies have shown that participants may make different decisions in a hypothetical scenario than they would in real life [20] [11].

Given these limitations I would not consider my results sufficient to form definitive conclusions on the performance of the systems. However, I do think they provide useful initial observations that could be explored further as part of a larger study addressing these issues, for example by conducting tests on a larger number of participants in vari-

ous real life situations using each of the different systems independently.

5.4 Analysis of User Testing Results

I carried out two user tests, the first with four participants and the second with three different participants. The following tables compare the output from each of the systems, calculated using the user input, with the actual decision made by the group following a group discussion. In each case, the entry shows whether the corresponding individual or the group is *For* or *Against* the initial claim.

Each row contains the results for a single user, except for the final row in each table which contains the results for the overall group. The *Value System* and *Preference System* columns show the output for the value-based system and the preference-based system respectively. For example in User Test 1, the first row shows that both the value-based system and the preference-based system determined that User 1 should be *For* the initial claim, whilst the final row shows that both systems determined that the group overall would be *Against* the initial claim. For the *Value System*, the number in brackets represents the calculated acceptance probability of the initial claim for each user, except for the last row which shows the average acceptance probability for the group. As a reminder, an acceptance probability above 0.5 is in favour of the initial claim, and an acceptance probability less than 0.5 is not in favour of the initial claim. For the *Preference System* column the individual users can only be either entirely *For* or *Against* the initial claim. The number in brackets in the last row therefore represents the proportion of users *For* the initial claim.

The *Voting System* column shows the output for the voting-based system. This is only calculated at the group level, so individual results are not given. The number in brackets represents the final weight of the initial claim. A final weight above 0.5 is in favour of the initial claim, and a final weight less than 0.5 is not in favour of the initial claim.

Finally, in the *Actual Decision* column, the last row shows the actual decision made by the group following a discussion (without seeing the output of any of the systems). The other rows show the view of each individual following the group discussion. For example in User Test 1 whilst each of the three systems determined the users would be *Against* the initial claim, the actual result of the group discussion was that they were *For* the initial claim.

It is worth remembering that the system's goal is not to predict the user decisions, but to help them make better group decisions. Therefore, differences between the system output and the actual group decision do not necessarily mean that the system isn't working well. These results need to be considered along with the actual content of the group discussions to evaluate the performance of each system, which I have done in the following section.

The full set of results including the user inputs and system outputs are included in Appendix F.

5.4.1 User Test 1

User	Value System	Preference System	Voting System	Actual Decision
User 1	For (0.86)	For	-	For
User 2	Against (0.00)	Against	-	Against
User 3	Against (0.00)	Against	-	For
User 4	Against (0.00)	Against	-	For
Group 1	Against (0.22)	Against (0.25)	Against (0.46)	For

Figure 22: *Results of the first group user test.*

5.4.2 User Test 2

User	Value System	Preference System	Voting System	Actual Decision
User 5	For (0.86)	For	-	For
User 6	For (0.61)	Against	-	Against
User 7	For (0.86)	Against	-	Against
Group 2	For (0.78)	Against (0.33)	0.For (0.59)	Against

Figure 23: *Results of the second group user test.*

5.5 Value-Based System Results

During the discussions the arguments expected to be most influential according to the system's individual evaluations were often different to the actual views put forward by those individuals during the discussion, more so than for the other systems. Furthermore, when considering the individual user output, the value-based system output only matched the actual decision of three of the seven participants. This suggests that the system's overall group decision was likely different from the group decision in both cases because the system had not accurately captured the subjective views of the group, rather than due to the system having made a better or more representative decision.

These concerns were apparent in the feedback on the system output collected by the users. Specifically, the users commented that the values promoted by the arguments were too blunt to be able to properly evaluate the nuances in the arguments. For some users there appeared to be a trade-off between different values. For example in this test, some arguments related to the value of 'human welfare including preservation of life' and others to 'societal advancement of knowledge and exploration'. However, even users who valued human welfare higher were generally comfortable with there being some risk to human life for sufficient scientific progress.

The users also commented that their ranking of the values might change depending on the scenario being considered. For my testing, users were asked to provide their value

preferences before knowing the context, however one user commented that had the context been provided first they might have changed their ranking. However, this goes against one of the proposed benefits of value-based approaches in that there are efficiencies from having a pre-agreed set of values and preferences that can be used when evaluating any set of arguments.

Furthermore, the users noted that the output from the value-based system was somewhat confusing given that any argument without an attacker was considered acceptable, resulting in most arguments seeming to be accepted by the group.

5.6 Preference-Based System Results

This time the arguments expected to be most influential according to the system's individual evaluations generally matched the actual views put forward by those individuals during the discussion. Moreover, the preference-based individual evaluations matched the actual decision of five of the seven participants. This suggests that the system was able to capture the subjective views of the participants quite accurately.

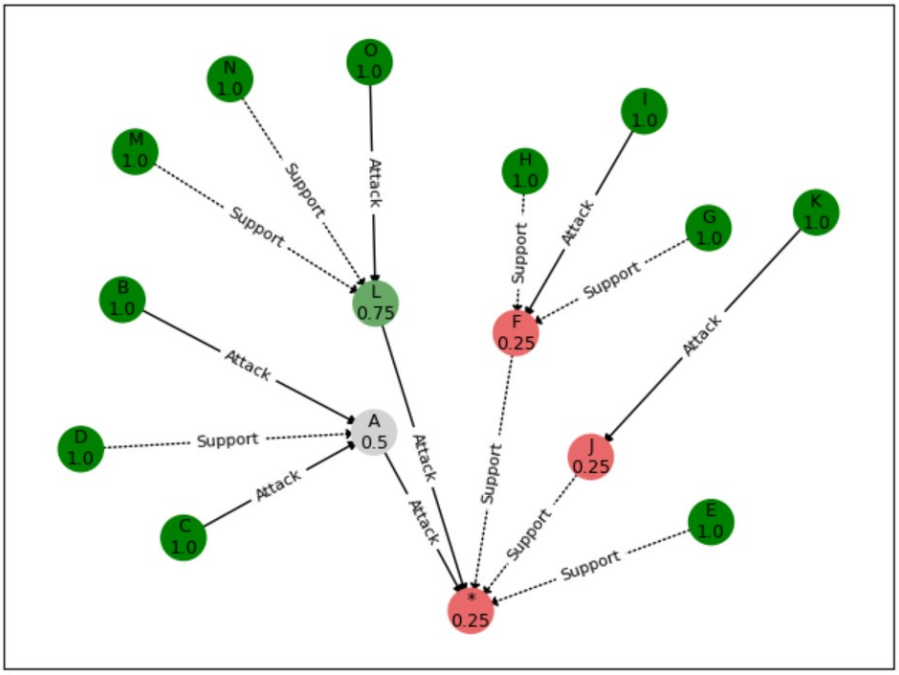


Figure 24: *Evaluation of Pref-BAF for Group 1.*

In the first user test, the system decision (see Figure 24 in which the initial claim is represented by '*') was different to the group's decision, which was in favour of the initial claim. This happened because the system expected three users to be against the initial claim, whilst actually two of those users were in favour of it. Based on the discussions, the decision for one of these users had been marginal. For the second user, the system had expected them to be against the initial claim due to their strong preference for one of its direct attackers, argument *A*, which the user preferred over its two attackers, arguments *B* and *C*. However, comments from the user suggested that the combined strength

of these two attackers should have defeated argument A, which my system is not able to model.

When a user was shown the system output it showed that generally the level of acceptance for arguments directly supporting the initial claim (F and J) was lower than for arguments directly attacking the initial claim (A and L). Whilst the user acknowledged this might warrant further discussion, they felt it was unlikely to change the group decision. In particular, they noted that argument B had been very influential in the group decision, which was an indirect supporter of the initial claim (in that it attacked A which attacked the initial claim). It is therefore possible that had the underlying argumentation framework been structured differently, with an argument similar to B directly attacking the initial claim it might have resulted in the system output matching the actual group decision.

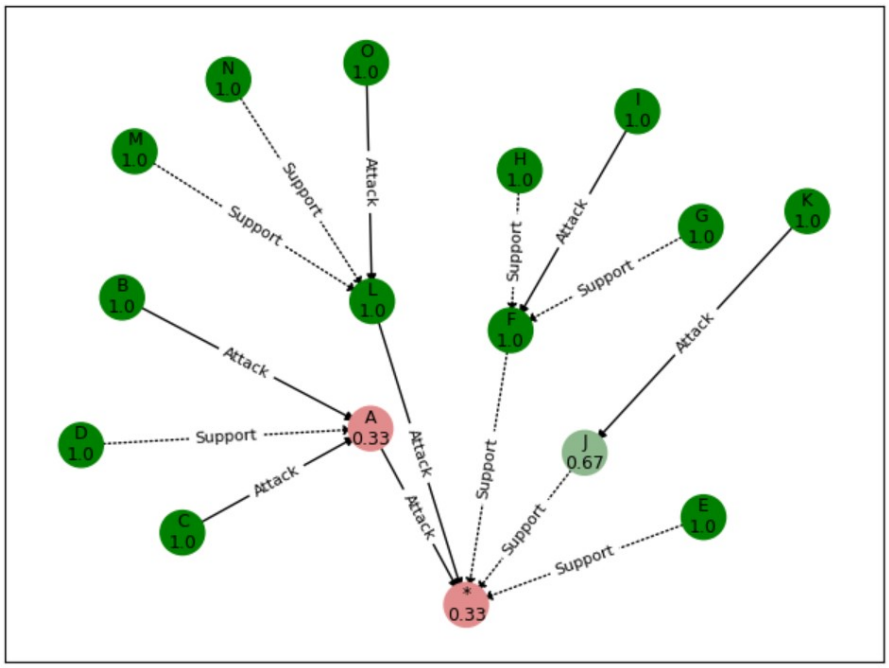


Figure 25: *Evaluation of Pref-BAF for Group 2.*

In the second user test, the system decision (see Figure 25) matched the group's decision, which was against the initial claim. This happened because each individual user system evaluation matched the actual decision of the corresponding user, and two of the three users were against the initial claim. However, the reasons why these two users were against the initial claim were different. Of the arguments directly connected to the initial claim, one user's most preferred argument was the attacking argument A and their least preferred argument was the attacking argument L, whilst the opposite was true for the other user as L was their most preferred argument and A their least preferred argument. However, this was not clear from the system output as, although the split in opinion on A was apparent, L appeared to be unanimously accepted. Interestingly, when the user was shown the output they did not pick up on this point, but instead they believed that

the system output reflected and therefore supported the group discussion. However, the user noted that the group had entirely dismissed *E* and that *F* was only somewhat accepted by the group, despite the system showing both as being unanimously accepted.

As with the value-based system, the users noted that the output from the preference-based system was also somewhat confusing because any argument without an attacker was considered unanimously acceptable.

5.7 Voting-Based System Results

For most of the arguments their final weights in the system evaluation broadly matched their perceived level of acceptance within the group discussions. There were a small number of arguments that appeared to be more influential in the group discussions than expected by the system, for example in the first user test *C* was noted as an important argument by three of the four users, despite only having a system weight of 0.5, corresponding to a neutral view. However, overall that the system seemed to capture the subjective views of the users quite accurately.

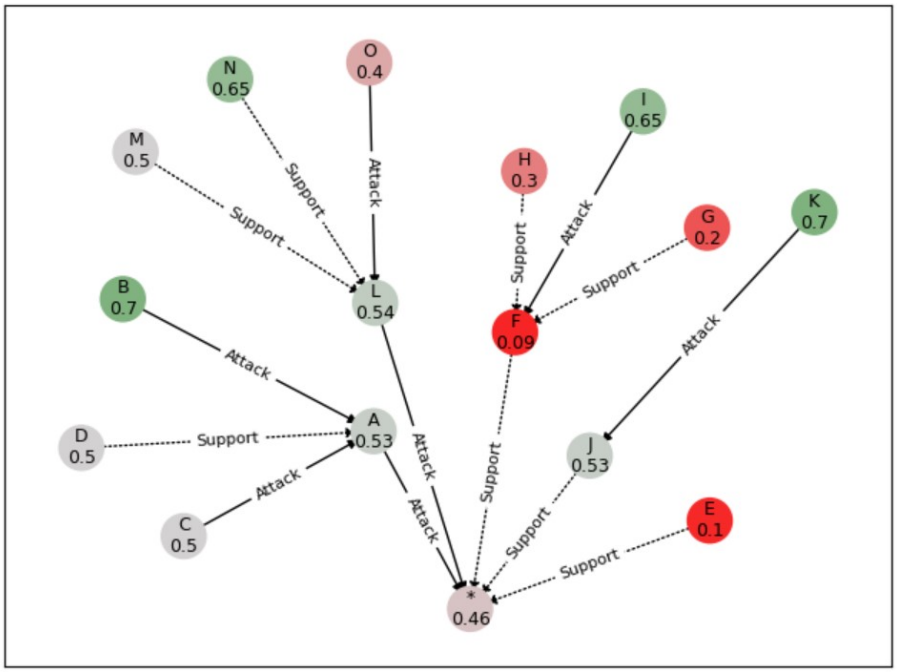


Figure 26: *Evaluation of Vote-BAF for Group 1.*

In the first user test, the system output (see Figure 26) was different to the group's decision, which was in favour of the initial claim. The system evaluation was against the initial claim because of the low final weights of most of the direct supporters of the initial claim, particularly *E* and *F*. Meanwhile, most of the group commented that arguments *B* and *C*, which were indirect supporters of the initial claim, were the key arguments for their decision. This suggests that a different framework structure could have changed the outcome, as with the preference-based system.

When a user was shown the system output, they commented that the support for the initial claim was lower than they had expected. They believed further discussion was warranted with regards to the direct supporting arguments of the initial claim, and whether the original decision could be justified given the lack of support for those arguments. That being said, the user felt the decision was unlikely to change given the strength of arguments *B* and *C*, which they felt were more influential than what the system output showed.

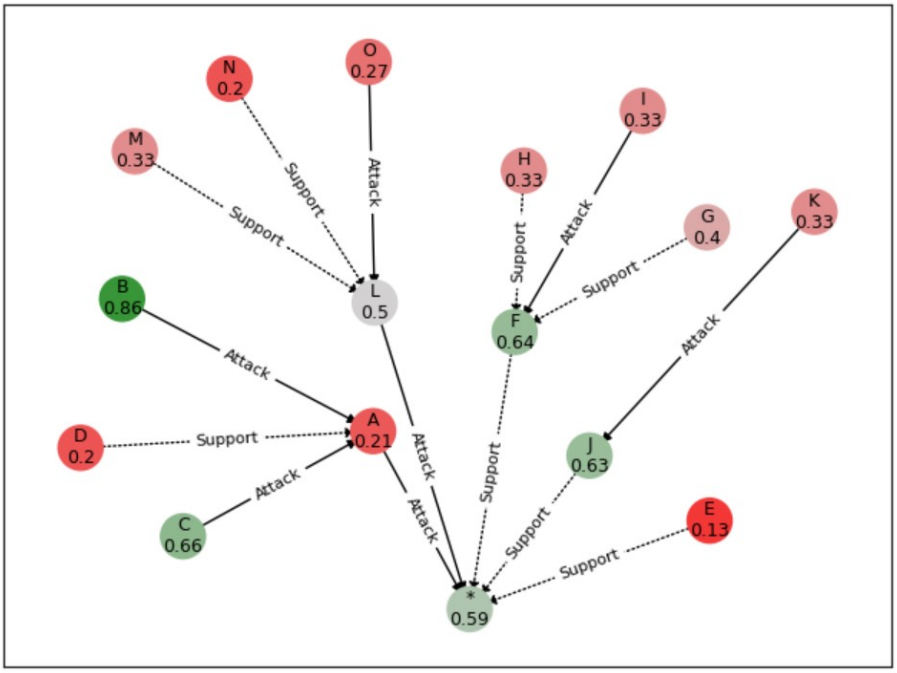


Figure 27: *Evaluation of Vote-BAF for Group 2.*

In the second user test, the system output (see Figure 27) was also different to the group's decision, which was against the initial claim. As explained in the previous section, the two users who were against the initial claim were for opposing reasons and so their votes in relation to the attacking arguments of the initial claim (*A* and *L*) worked to offset each other. Meanwhile the arguments in support of the initial claim generally scored higher and so the system was in favour of the initial claim.

When a user was shown the system output, they seemed surprised that the final weights of the initial claim's attackers were so low. When the reasons for this were explained to them, they stated that if given the opportunity they would revisit the original group decision to determine whether it was still justified despite the disagreement on the arguments attacking the initial claim. This is an interesting point more generally, and whether the 'best' decision is the one favoured by most users or the one where there is the most general acceptance among the underlying supporting arguments will vary depending on the situation.

In terms of output, the users unanimously agreed that the voting-based system output was the easiest to interpret, especially as arguments without attackers could have a low

weight.

5.8 General Observations

More generally, in both tests the users did appear to make their decision primarily based on the arguments presented to them. Where the users put forward new arguments, these were mostly counterarguments to the arguments that had been presented to them. For instance argument E was generally perceived as a weak argument and users came up with new arguments that attacked it, even though no arguments attacking E were given to the users initially. Overall, I do not believe that the difference in output between the systems and the group discussion was due to the users putting forward new arguments.

5.9 Evaluation

Considering both the evaluation against the specifications and the user testing, the value-based system appears to be the weakest of the three systems. Whilst it requires the least onerous user input and it scales well to larger frameworks, the user testing showed that it was often unable to differentiate between nuances in arguments and as a result users were generally unconvinced by its output, which did not seem to reflect their subjective beliefs. Furthermore, it is the most susceptible to being gamed of the three systems.

Evaluating whether the preference-based or voting-based system works better is difficult based on my small sample of results. Both broadly meet the system specifications, although the voting-based system is more susceptible to being gamed and can give users with more polarised views slightly more influence on the outcome. However, users believed the voting-based system output was easier to understand and better highlighted the most influential arguments. This is supported by the fact that in both user tests, the users stated that the voting-based system output would warrant further discussion (even if they felt it was unlikely to change the decision).

Interestingly, the tests demonstrated these two systems can produce very different output. This is because the preference-based system output is determined by the most preferred arguments of each user, whilst the voting-based system takes a more balanced approach to considering the general acceptance of each argument amongst all users. Which methodology is most useful to users will likely depend on the decision being discussed and further research could investigate this further.

6 Conclusion

Within this project I have developed three new systems, each using different subjective input, that can be used to help groups make decisions through arguments. I did this by combining and extending existing argumentation frameworks. I believe my most significant contributions have been to determine: i) a methodology to evaluate preference-based and value-based argumentation frameworks within a bipolar setting, ii) the most appropriate way to aggregate the views of multiple users in these two frameworks, and iii) how to extend existing voting frameworks to allow for weighted votes.

Having evaluated these systems against the desired specifications and through user testing I concluded that there was no perfect system. Whilst each system generally met most of the desired system specifications each also had their own drawbacks. Given the limited user data collected, I would be reluctant at this stage to draw any definitive conclusions on the performance of any system.

However, based on my results the value-based system appeared to be the least suitable system given its issues in accurately reflecting users' subjective views and its greater susceptibility to being gamed. On the other hand, the preference-based and voting-based systems can better defend against system abuse, more accurately reflected the user subjective views, and provided more insightful output during the user testing. As such, both systems showed promise in achieving the main aim of helping groups to make decisions. In terms of whether one system is better than the other, my analysis showed each had different pros and cons. Combined with the fact that the systems can also have different outcomes, further testing is needed to determine which system is most useful for a given situation, or whether groups might benefit from using both systems.

6.1 Possible Areas of Further Study and Testing

An obvious area for further work would be to gather more results for my three systems, addressing the limitations discussed in Section 5.3, to enable a more informed analysis. However, beyond that there are several areas of possible future study:

- Developing and evaluating systems that use other argumentation frameworks, such as the meta-structures of Extended Argumentation Frameworks or alternative quantitative frameworks. Or investigating alternative design choices for my systems, for example the ranking method I outlined for determining acceptable sets of arguments in the preference-based system.
- Considering how my systems could be adapted to enable evaluation of cyclic frameworks, noting their current restriction to acyclic frameworks.
- Finally, a more ambitious study could try to assess the long-term impacts on individuals regularly using decision-making systems in order to determine safeguards against the potential issues I discuss in the next section.

7 Legal, Social, Ethical & Professional Issues

7.1 Legal, Social & Ethical Issues

An important issue to address with regards to using AI for decision-making is liability. Consider a situation in which a government uses my system to help them make a decision, but that decision has catastrophic consequences. Who should be held accountable - the government using the system, the system developers or the system itself?

Whilst there has been much discussion over whether an AI could be granted its own legal personality, which would open it up to liability in a similar way that companies can be held liable, the current legal consensus is against this approach [13] [15]. However, even assigning liability between developers and users can be difficult given the complexity of many AI products, which makes it difficult to apportion blame amongst the various parties involved in making the product. Demonstrating the scale of the challenge, the UK government's recent white paper on AI regulation stated it is *"too soon to make decisions about liability as it is a complex, rapidly evolving issue"* [22].

There are also questions over the extent to which AI should be allowed to make decisions. In the introduction I explained that my system should only aid in decision-making, rather than be the decision-maker. This broadly mirrors current UK law in which certain decisions with potentially significant effects cannot be automated. Specifically, Article 22(1) of the UK GDPR states *"The data subject shall have the right not to be subject to a decision based solely on automated processing, including profiling, which produces legal effects concerning him or her or similarly significantly affects him or her"* [35].

However, as society becomes increasingly comfortable with and dependent on AI there is a risk that AI will be allowed to make larger and more important decisions with little to no human oversight. It could even become the case that humans are questioned, and potentially legal action taken against them, for not following the recommended decision of the AI system. For example, imagine a doctor who uses a decision-making AI to help diagnose a patient. Even if they disagree with the assessment of the AI they may decide to follow its recommendation if they know they may be sued if they ignore its advice and turn out to be wrong [40].

Such decision-systems are likely to have a wider impact on society. Were my system, or similar products, to be adopted more generally and used as intended, then improved decision-making could benefit many areas of society, from governments to businesses to educational settings. The reasoning skills of individuals could improve as the system provides feedback to the users and allows them to reassess potential biases. However, there is also a significant risk that the system is relied upon to make decisions rather than to support decision-making. This could have the opposite intended effect, with the population experiencing overall cognitive decline as more and more decisions are outsourced to AI systems.

Such a deference to AI decision-makers could also have important ethical impacts relat-

ing to human autonomy and self-determination [27]. As individuals make fewer decisions many may start to believe they have less control over their lives. It has been argued that this feeling of autonomy is a crucial part of human well-being [45] and dignity [26]. That is not to say that AI should never be allowed to make decisions, but that further thought needs to be given over how and where it is allowed to do so. Of course, a principle argument in favour of AI is that it could make our mundane decisions for us, freeing up time for humanity to focus on larger issues.

7.2 Professional Issues

Whilst there is not yet a universally accepted professional code for the development of AI systems, the Asilomar AI Principles are a set of 23 principles proposed by the Future of Life Institute that aim to promote the safe and beneficial development of AI technology [34]. I believe that if used as intended my system adheres to these principles. The following principles being especially relevant:

- *Research Goal: The goal of AI research should be to create not undirected intelligence, but beneficial intelligence.* The 'intelligence' within my system is used to help groups make better decisions than they would have otherwise. A particular motivation of mine has been the way in which governments make decisions, and many people would benefit from better government decisions.
- *Failure Transparency: If an AI system causes harm, it should be possible to ascertain why.* The underlying methodology of my system is significantly more transparent than other popular 'black-box' AI methods, notably neural networks. Specifically, my system outputs an auditable argumentation graph highlighting the most influential arguments.
- *Human Values: AI systems should be designed and operated so as to be compatible with ideals of human dignity, rights, freedoms, and cultural diversity.* This is directly addressed by the value-based version of my system, where decisions are made according to the preferred values of users. These ideals are also captured indirectly by the preference-based and voting-based systems through the subjective user input.
- *Human Control: Humans should choose how and whether to delegate decisions to AI systems, to accomplish human-chosen objectives.* The way in which AI decision-making systems are used could have profound effects on humanity. My intention is that decisions are not entirely delegated to my system but that it is used as a tool to help humans make decisions.

On a more practical level, I have developed my systems according to the principles set out in the Code of Conduct & Code of Good Practice issued by the British Computer Society [1].

Appendix

A Acceptance Probability in a Val-BAF

As per the main body of my report, first I defined the following sets:

- Let $att(A)$, $sup(A)$ and $pre(A)$ denote the sets containing the attackers of A, the supporters of A, and the union of these two sets, respectively.
- Let $pre^{v}(A) = \{B \in AR | val(B) = v, B \in pre(A)\}$.
- Let $pre^{v+}(A) = \{B \in AR | val(B) >> v, B \in pre(A)\}$.
- Let 2^{Y} denote the powerset of set Y.

Then for my value-based system I defined the acceptance probability for an argument as follows:

Definition (Acceptance Probability in a Val-BAF): The acceptance probability for an argument in a Val-BAF is a function defined as $f{:}AR \mapsto [0,1]$ where, for any $A \in AR$:

$$p_{acc}(A) = 1 - \sum_{\substack{v \in V, \\ s.t.\ v \gg val(A) \\ or\ v = val(A)}} \sum_{\omega \in 2^{pre^{v}(A)}} \left(\prod_{a \in \omega} (p_{acc}(a)) \cdot \prod_{b \in pre^{v}(A)/\omega} (1 - p_{acc}(b)) \cdot \prod_{c \in pre^{v+}(A)} (1 - p_{acc}(c)) \cdot \frac{|att(A) \cap \omega|}{|\omega|} \right)$$

where $|\omega|$ and $|att(A) \cap \omega|$ are the total number of arguments in ω and in the intersection of ω and $att(A)$ respectively.

As noted in the main body of the report, for each value which is the same as or more preferred than the value of A, the function considers possible combinations of arguments promoting that value being acceptable or not. A specific combination of acceptable arguments promoting a given value is represented by ω, arguments in this set are accepted and arguments promoting the given value that are not in the set are not accepted. The formula in the definition calculates the probability of this combination occurring - the first product calculates the probability that the arguments in ω are accepted and the second product calculate the probability that the arguments not in ω are not accepted. The third product then calculates the probability that all the arguments promoting more preferred values are not accepted, for if at least one of them was accepted then the arguments promoting less preferred values would become irrelevant.

The final term on the right then calculates the proportion of attacking arguments in ω to determine the probability that the attack would be successful if the specific combination of arguments in ω occured. It assumes that for arguments promoting the same value they all have an equal chance of being the strongest argument.

The second summation means that for a given value every possible combination of arguments promoting that value being accepted and not accepted is considered, whilst the first summation sign means that this calculation is carried out for every value in the framework which is the same or more preferred than the value of A. There is no need to check for values that are less preferred than the value of A as A will always be able to defend itself against any attackers promoting that value.

Under this definition if an argument A has no attackers, or if $val(A)$ is preferred to the values of all of its attackers, or if A has a supporter C with $p_{acc}(C) = 1$ and $val(C)$ is preferred to the value of all of A's attackers then $p_{acc}(A) = 1$. If there is an attacker B with $p_{acc}(B) = 1$ and $val(B)$ is preferred to or is the same as $val(A)$ and also preferred to the value of any of A's supporters then $p_{acc}(A) = 0$. In other words if there is a single argument promoting the most preferred value it will always win, akin to the method used for Pref-BAFs in which the most preferred argument always won.

If A has at least one attacker and one supporter all promoting the same value, and that value is the same as or more preferred than $val(A)$, then we assume each of the attackers and supporters has an equal probability of being the strongest argument. Therefore, the acceptance probability of A now lies somewhere between 0 and 1. For example, in a framework consisting of five arguments: the initial claim A with one attacker and three supporters, with all arguments promoting the same value, my system assumes there is a 25% chance that the attacker is the strongest argument and so a 25% chance that A will be defeated by it, therefore $p_{acc}(A) = 0.75$. The function also adjusts for the acceptance probability of these supporters and attackers. In the previous example if the attacker had an acceptance probability of 0.5, and the supporters all had an acceptance probability of 1, then the probability of the attacker being both accepted and the strongest argument is now 12.5%, so $p_{acc}(A) = 0.875$.

B Preference-Based System Source Code

The following pages contain the source code for my Preference-Based System and an example evaluation using the argumentation framework from the test case in Appendix E.

The code has been written in Python 3.9, and makes use of the following open-source non-standard libraries:

- Matplotlib: *a comprehensive library for creating static, animated, and interactive visualizations in Python.* [29]
- NetworkX: *a package for the creation, manipulation, and study of the structure, dynamics, and functions of complex networks.* [32]

I verify that I am the sole author of the programmes contained in this appendix, except where explicitly stated to the contrary.

William Feeney. 11 August 2023.

```
# The code below implements my Preference-Based System. The argumentation
framework can be changed
# by changing the arguments, values, attacks and supports towards the end
of the code.  The argument
# preferences can also be changed and/or additional users added at the end
of the code.


import networkx as nx
import matplotlib.pyplot as plt
from matplotlib.colors import LinearSegmentedColormap

# Creates a class for the arguments, containing information on their id
(typically a letter), the arguments
# attacking it, the arguments supporting it, where it is accepted or not
and the proportion of users
# for which it is accepted. NB the acceptance the group acceptance
proportion are  initially set to 1
# and 0 respectively, these are updated when the evaluation functions are
run.


class Argument:
    def __init__(self, id):
        self.id = id
        self.attacks = set()
        self.supports = set()
        self.acceptance = 1
        self.group_acceptance_prop = 0

    # Calculate if an argument is accepted for a given user's preferences.
A value of 1 means the argument
    # is accepted, and 0 otherwise.
    def calculate_acceptance(self, args, user_pref):
        # If an argument has no attackers it will be accepted
        if self.attacks == set():
            return 1

        # If an argument has an attacker which is more preferred it will
not be accepted
        elif self.attacks != set() and self.supports == set():
            for attack in self.attacks:
                for argument in args:
                    if attack == argument.id:
                        if user_pref[argument.id] >= user_pref[self.id]:
                            return 0
            return 1

        # If an argument has an supporter which is more preferred than any
attacker then the argument
        # will be accepted
        else:
            most_pref_arg = self.id
            for argument in self.attacks + self.supports:
                for arg in args:
                    if argument == arg.id:
                        if user_pref[arg.id] > user_pref[most_pref_arg] and
arg.calculate_acceptance(args, user_pref) > 0:
                            most_pref_arg = arg.id
            if most_pref_arg in self.supports or most_pref_arg == self.id:
                return 1
```

```
            else:
                return 0


# For a specified argument, returns the arguments that attack it.
def get_attacks(args):
    attacks = set()
    for argument in args:
        for attack in argument.attacks:
            attacks.add((attack, argument.id))
    return attacks


# For a specified argument, returns the arguments that support it.
def get_supports(args):
    supports = set()
    for argument in args:
        for support in argument.supports:
            supports.add((support, argument.id))
    return supports


# Calculates whether each of the arguments is accepted or not for a single
user.
def calculate_individual_acceptance(args, user_pref):
    for argument in args:
        argument.acceptance = argument.calculate_acceptance(args,
user_pref)


# For each argument, calculates the proportion of users that find it
acceptable.
def calculate_group_average_acceptance(args, user_preferences):
    for user in user_preferences:
        calculate_individual_acceptance(args, user)

        for argument in args:
            argument.group_acceptance_prop += argument.acceptance /
len(user_preferences)


# Creates a graph representation of the preference bipolar argumentation
framework for an individual user.
def create_individual_pref_baf_graph(args):
    graph = nx.DiGraph()

    # Add arguments as nodes
    for argument in args:
        graph.add_node(argument.id, acceptance=argument.acceptance)

    # Add attacks as directed edges
    attacks = get_attacks(args)
    for attack in attacks:
        graph.add_edge(attack[0], attack[1], linestyle='solid')

    # Add supports as directed edges
    supports = get_supports(args)
    for support in supports:
        graph.add_edge(support[0], support[1], linestyle='dotted')

    return graph
```

```
# Creates a graphical representation of the preference bipolar
argumentation framework for an individual user.
def visualize_individual_pref_baf_graph(graph):
    pos = nx.spring_layout(graph, iterations=10000, seed=15) # Seed can be
changed for other frameworks

    edge_styles = [graph[u][v]['linestyle'] for u, v in graph.edges()]
    node_weights = list(nx.get_node_attributes(graph,
'acceptance').values())
    node_labels = {node: f"{node}" for node in graph.nodes()}
    edge_labels = {(u, v): "Attack" if graph[u][v]['linestyle'] == 'solid'
else "Support" for u, v in graph.edges()}

    plt.figure(figsize=(8, 6))

    # Creates a colour map for the nodes based on whether they are accepted
or not
    cmap = LinearSegmentedColormap.from_list(
        'custom_cmap',
        [(0, 'red'), (0.5, 'lightgrey'), (1, 'green')],
        N=256
    )

    # Adds the nodes, node label, edges and edge labels
    nx.draw_networkx_nodes(graph, pos, node_color=node_weights, cmap=cmap,
vmin=0, vmax=1, node_size=500)
    nx.draw_networkx_labels(graph, pos, font_size=9, labels=node_labels)
    nx.draw_networkx_edges(graph, pos, edge_color='black',
style=edge_styles, arrows=True, arrowsize=10)
    nx.draw_networkx_edge_labels(graph, pos, edge_labels=edge_labels,
font_color='black', font_size=8)

    plt.axis('on')
    plt.show()


# Creates a graph representation of the group value bipolar argumentation
framework.
def create_group_pref_baf_graph(args):
    graph = nx.DiGraph()

    # Add arguments as nodes
    for argument in args:
        graph.add_node(argument.id,
group_acceptance_prop=argument.group_acceptance_prop)

    # Add attacks as directed edges
    attacks = get_attacks(args)
    for attack in attacks:
        graph.add_edge(attack[0], attack[1], linestyle='solid')

    # Add supports as directed edges
    supports = get_supports(args)
    for support in supports:
        graph.add_edge(support[0], support[1], linestyle='dotted')

    return graph
```

```
# Creates a graphical representation of the group preference bipolar
argumentation framework.
def visualize_group_pref_baf_graph(graph):
    pos = nx.spring_layout(graph, iterations=10000, seed=15) # Seed can be
changed for other frameworks

    edge_styles = [graph[u][v]['linestyle'] for u, v in graph.edges()]
    node_weights = list(nx.get_node_attributes(graph,
'group_acceptance_prop').values())
    node_labels = {node:
f"{node}\n{str(graph.nodes[node]['group_acceptance_prop'])}" for node in
graph.nodes()}
    edge_labels = {(u, v): "Attack" if graph[u][v]['linestyle'] == 'solid'
else "Support" for u, v in graph.edges()}

    plt.figure(figsize=(8, 6))

    # Creates a colour map for the nodes based on whether they are accepted
or not.
    cmap = LinearSegmentedColormap.from_list(
        'custom_cmap',
        [(0, 'red'), (0.5, 'lightgrey'), (1, 'green')],
        N=256
    )

    # Adds the nodes, node labels, edges and edge labels
    nx.draw_networkx_nodes(graph, pos, node_color=node_weights, cmap=cmap,
vmin=0, vmax=1, node_size=500)
    nx.draw_networkx_labels(graph, pos, font_size=9, labels=node_labels)
    nx.draw_networkx_edges(graph, pos, edge_color='black',
style=edge_styles, arrows=True, arrowsize=10)
    nx.draw_networkx_edge_labels(graph, pos, edge_labels=edge_labels,
font_color='black', font_size=8)

    plt.axis('on')
    plt.show()


# Evaluates the group preference bipolar argumentation framework and shows
its graphical representation.
def evaluate_group_pref_baf(args, user_preferences):
    calculate_group_average_acceptance(args, user_preferences)
    graph = create_group_pref_baf_graph(args)
    visualize_group_pref_baf_graph(graph)


# Evaluates the preference bipolar argumentation framework and shows its
graphical representation for an individual
# user.
def evaluate_individual_pref_baf(args, user_preferences):
    calculate_individual_acceptance(args, user_preferences)
    graph = create_individual_pref_baf_graph(args)
    visualize_individual_pref_baf_graph(graph)


# Define the arguments
arguments = [
    Argument('*'),
    Argument('A'),
    Argument('B'),
    Argument('C'),
```

```
    Argument('D'),
    Argument('E'),
    Argument('F'),
    Argument('G'),
    Argument('H'),
    Argument('I'),
    Argument('J'),
    Argument('K'),
    Argument('L'),
    Argument('M'),
    Argument('N'),
    Argument('O'),
]

# Define the attacks
arguments[0].attacks = ['A', 'L']
arguments[1].attacks = ['B', 'C']
arguments[6].attacks = ['I']
arguments[10].attacks = ['K']
arguments[12].attacks = ['O']

# Define the supports
arguments[0].supports = ['E', 'F', 'J']
arguments[1].supports = ['D']
arguments[6].supports = ['H', 'G']
arguments[12].supports = ['M', 'N']


# Define the argument preference orderings for each user (the higher the
number the more preferred).
# NB the actual numbers used are not important just the relative ranking
for each user.
user_preferences = [
    {'*': 0, 'A': 3, 'B': 6, 'C': 5, 'D': 4, 'E': 4, 'F': 2, 'G': 3, 'H':
4, 'I': 5,
    'J': 5, 'K': 6, 'L': 1, 'M': 3, 'N': 4, 'O': 2},

    {'*': 0, 'A': 5, 'B': 3, 'C': 4, 'D': 2, 'E': 1, 'F': 2, 'G': 4, 'H':
3, 'I': 5,
    'J': 4, 'K': 5, 'L': 3, 'M': 2, 'N': 4, 'O': 5},

    {'*': 0, 'A': 3, 'B': 5, 'C': 2, 'D': 4, 'E': 1, 'F': 2, 'G': 3, 'H':
5, 'I': 4,
     'J': 4, 'K': 3, 'L': 5, 'M': 3, 'N': 4, 'O': 2},

    {'*': 0, 'A': 5, 'B': 3, 'C': 2, 'D': 4, 'E': 1, 'F': 1, 'G': 2, 'H':
3, 'I': 4,
    'J': 3, 'K': 4, 'L': 4, 'M': 6, 'N': 3, 'O': 5}
    ]

evaluate_individual_pref_baf(arguments, user_preferences[0])
evaluate_group_pref_baf(arguments, user_preferences)
```

Example Individual User Evaluation

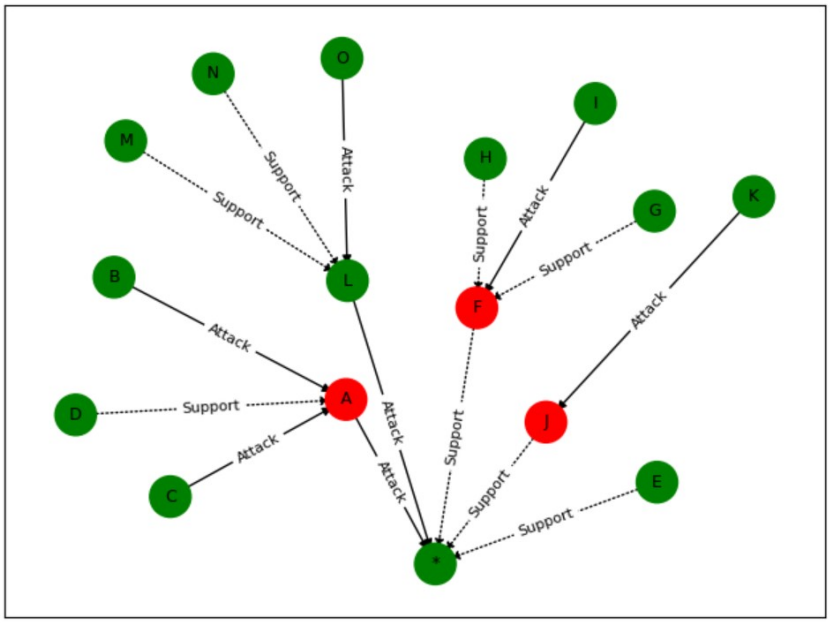


Figure 28: *An example evaluation output of the Preference-Based System for an individual user.*

The output shows that the user is in favour of the initial claim (*).

Example Group Evaluation

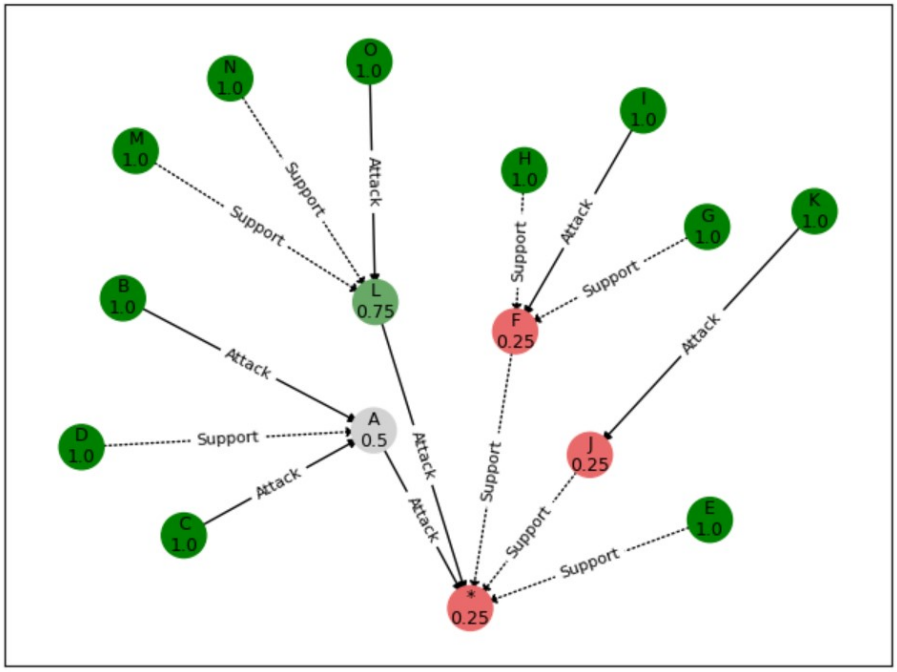


Figure 29: *An example evaluation output of the Preference-Based System for a group.*

The output shows that the group is not in favour of the initial claim (*), with only 25% of the inidividual user frameworks in favour of it.

C Value-Based System Source Code

The following pages contain the source code for my Value-Based System and an example evaluation using the argumentation framework from the test case in Appendix E.

The code has been written in Python 3.9, and makes use of the following open-source non-standard libraries:

- Matplotlib: *a comprehensive library for creating static, animated, and interactive visualizations in Python.* [29]
- NetworkX: *a package for the creation, manipulation, and study of the structure, dynamics, and functions of complex networks.* [32]

I verify that I am the sole author of the programmes contained in this appendix, except where explicitly stated to the contrary.

William Feeney. 11 August 2023.

```
# The code below implements my Value-Based System. The argumentation
framework can be changed
# by changing the arguments, values, attacks and supports towards the end
of the code.  The value
# preferences can also be changed and/or additional users added at the end
of the code.


import networkx as nx
import matplotlib.pyplot as plt
from matplotlib.colors import LinearSegmentedColormap
import itertools

# Creates a class for the arguments, containing information on their id
(typically a letter), the arguments
# attacking it, the arguments supporting it, its value, its acceptance
probability and its group average
# acceptance probability. NB the acceptance probability and the group
average acceptance probability are
# initially set to 1 and 0 respectively, these are updated when the
evaluation functions are run.
class Argument:
    def __init__(self, id):
        self.id = id
        self.attacks = set()
        self.supports = set()
        self.value = '*'
        self.accept_prob = 1
        self.group_accept_prob = 0

    # Calculate the acceptance probability of an argument for a given
user's value preferences.
    def calculate_acceptance_probability(self, args, user_value_pref):
        # If an argument has no attackers it will have an acceptance
probability of 1
        if self.attacks == set():
            return 1

        # Find the values that are the same as or more preferred than the
value of the argument.
        all_values = get_values(args)
        arg_value = self.value

        preferred_values = []
        for value in all_values:
            if user_value_pref[value] >= user_value_pref[arg_value]:
                preferred_values.append(value)

        # Find the arguments associated with each of those values, and
store them in 'pre'.
        pre = {}

        for i in range(len(preferred_values)):
            curr_value = preferred_values[i]

            temp = []
            for argument in arguments:
                if argument.value == curr_value and argument.id in
self.attacks:
                    temp.append(argument.id)
```

```
                elif argument.value == curr_value and argument.id in
self.supports:
                    temp.append(argument.id)

            pre[curr_value] = temp

        # Function that returns the powerset of a set.
        def powerset(s):
            power_set = []
            for r in range(len(s) + 1):
                combinations = itertools.combinations(s, r)
                power_set.extend(combinations)
            return power_set

        # Calculate the acceptance probability of the argument by
calculating the probability of
        # combinations of acceptable and not acceptable arguments, and the
probability that under
        # that combination the argument is defeated.
        prob_successful_attack = 0

        for value in preferred_values:
            # Consider every possible combination of arguments with a given
value being accepted or not.
            # This is done by considering the power set of pre[value].
Arguments in this set represent
            # acceptable arguments associated with the given value.
            for omega in powerset(pre[value]):

                # Calculate the probability that all of the arguments in
omega are acceptable.
                if len(omega) == 0:
                    acc_prob = 0

                else:
                    acc_prob = 1
                    for a in omega:
                        for argument in args:
                            if argument.id == a:
                                acc_prob *=
argument.calculate_acceptance_probability(args, user_value_pref)

                # Calculate the probability that all of the arguments with
the current value being considered
                # and not in omega are not acceptable.
                pre_minus_acc = set(pre[value]).difference(omega)

                if len(pre_minus_acc) == 0:
                    not_acc_prob = 1

                else:
                    not_acc_prob = 1
                    for b in pre_minus_acc:
                        for argument in args:
                            if argument.id == b:
                                not_acc_prob *= (1 -
argument.calculate_acceptance_probability(args, user_value_pref))

                # # Calculate the probability that all of the arguments
with a more preferred value than
                # the current value being considered are not acceptable.
```

```
                not_acc_prob_val_plus = 1

                for argument in args:
                    if argument.id in self.attacks and
user_value_pref[argument.value] > user_value_pref[value]:
                        not_acc_prob_val_plus *= (1 -
argument.calculate_acceptance_probability(args, user_value_pref))

                for argument in args:
                    if argument.id in self.supports and
user_value_pref[argument.value] > user_value_pref[value]:
                        not_acc_prob_val_plus *= (1 -
argument.calculate_acceptance_probability(args, user_value_pref))

                # Calculate the proportion of acceptable arguments that are
attacks for the value being considered
                if len(omega) == 0:
                    att_prop = 0
                else:
                    att_prop = len(set(omega).intersection(self.attacks)) /
len(omega)

                # Multiply all the probabilities by the proportion of
acceptable attacks to calculate
                # the probability that the combination results in the
argument being defeated.
                overall_prob = acc_prob * not_acc_prob *
not_acc_prob_val_plus * att_prop

                # Add the probabilities for every possible combination
together to determine
                # the overall total probability of a successful attack
                prob_successful_attack += overall_prob

        # The acceptance probability is then 1 - probability of a
successful attack.
        return 1 - prob_successful_attack


# For a specified argument, returns the arguments that attack it.
def get_attacks(args):
    attacks = set()
    for argument in args:
        for attack in argument.attacks:
            attacks.add((attack, argument.id))
    return attacks


# For a specified argument, returns the arguments that support it.
def get_supports(args):
    supports = set()
    for argument in args:
        for support in argument.supports:
            supports.add((support, argument.id))
    return supports


# For a specified argument, returns the value associated with it.
def get_values(args):
    values = set()
    for argument in args:
```

```
        values.add(argument.value)
    return values


# Calculates the acceptance probability of all arguments for a single user.
def calculate_individual_acceptance_probability(args, user_value_pref):
    for argument in args:
        argument.accept_prob =
argument.calculate_acceptance_probability(args, user_value_pref)


# Calculates the average acceptance probability of all arguments for all
users.
def calculate_group_average_acceptance_probability(args,
user_value_preferences):
    for user in user_value_preferences:
        calculate_individual_acceptance_probability(args, user)

        for argument in args:
            argument.group_accept_prob += argument.accept_prob /
len(user_value_preferences)


# Creates a graph representation of the value bipolar argumentation
framework for an individual user.
def create_individual_value_baf_graph(args):
    graph = nx.DiGraph()

    # Add arguments as nodes
    for argument in args:
        graph.add_node(argument.id, value=argument.value,
accept_prob=argument.accept_prob)

    # Add attacks as directed edges
    attacks = get_attacks(args)
    for attack in attacks:
        graph.add_edge(attack[0], attack[1], linestyle='solid')

    # Add supports as directed edges
    supports = get_supports(args)
    for support in supports:
        graph.add_edge(support[0], support[1], linestyle='dotted')

    return graph


# Creates a graphical representation of the value bipolar argumentation
framework for an individual user.
def visualize_individual_value_baf_graph(graph):
    pos = nx.spring_layout(graph, iterations=10000, seed=15)  # Seed can be
changed for other frameworks

    edge_styles = [graph[u][v]['linestyle'] for u, v in graph.edges()]
    node_weights = list(nx.get_node_attributes(graph,
'accept_prob').values())
    node_labels = {node:
f"{node}\n{(round(graph.nodes[node]['accept_prob'],2))}" for node in
graph.nodes()}
    edge_labels = {(u, v): "Attack" if graph[u][v]['linestyle'] == 'solid'
else "Support" for u, v in graph.edges()}
```

```
    plt.figure(figsize=(8, 6))

    # Creates a colour map for the nodes based on their acceptance
probability
    cmap = LinearSegmentedColormap.from_list(
        'custom_cmap',
        [(0, 'red'), (0.5, 'lightgrey'), (1, 'green')],
        N=256
    )

    # Adds the nodes, node label, edges and edge labels
    nx.draw_networkx_nodes(graph, pos, node_color=node_weights, cmap=cmap,
vmin=0, vmax=1, node_size=500)
    nx.draw_networkx_labels(graph, pos, font_size=9, labels=node_labels)
    nx.draw_networkx_edges(graph, pos, edge_color='black',
style=edge_styles, arrows=True, arrowsize=10)
    nx.draw_networkx_edge_labels(graph, pos, edge_labels=edge_labels,
font_color='black', font_size=8)

    plt.axis('on')
    plt.show()


# Creates a graph representation of the group value bipolar argumentation
framework.
def create_group_value_baf_graph(args):
    graph = nx.DiGraph()

    # Add arguments as nodes
    for argument in args:
        graph.add_node(argument.id,
group_accept_prob=argument.group_accept_prob)

    # Add attacks as directed edges
    attacks = get_attacks(args)
    for attack in attacks:
        graph.add_edge(attack[0], attack[1], linestyle='solid')

    # Add supports as directed edges
    supports = get_supports(args)
    for support in supports:
        graph.add_edge(support[0], support[1], linestyle='dotted')

    return graph


# Creates a graphical representation of the group value bipolar
argumentation framework.
def visualize_group_value_baf_graph(graph):
    pos = nx.spring_layout(graph, iterations=10000, seed=15)  # Seed can be
changed for other frameworks

    edge_styles = [graph[u][v]['linestyle'] for u, v in graph.edges()]
    node_weights = list(nx.get_node_attributes(graph,
'group_accept_prob').values())
    node_labels = {node:
f"{node}\n{str(round(graph.nodes[node]['group_accept_prob'],2))}" for node
in graph.nodes()}
    edge_labels = {(u, v): "Attack" if graph[u][v]['linestyle'] == 'solid'
else "Support" for u, v in graph.edges()}
```

```
    plt.figure(figsize=(8, 6))

    # Creates a colour map for the nodes based on their acceptance
probability
    cmap = LinearSegmentedColormap.from_list(
        'custom_cmap',
        [(0, 'red'), (0.5, 'lightgrey'), (1, 'green')],
        N=256
    )

    # Adds the nodes, node labels, edges and edge labels
    nx.draw_networkx_nodes(graph, pos, node_color=node_weights, cmap=cmap,
vmin=0, vmax=1, node_size=500)
    nx.draw_networkx_labels(graph, pos, font_size=9, labels=node_labels)
    nx.draw_networkx_edges(graph, pos, edge_color='black',
style=edge_styles, arrows=True, arrowsize=10)
    nx.draw_networkx_edge_labels(graph, pos, edge_labels=edge_labels,
font_color='black', font_size=8)

    plt.axis('on')
    plt.show()


# Evaluates the group value bipolar argumentation framework and shows its
graphical representation.
def evaluate_group_val_baf(args, user_value_preferences):
    calculate_group_average_acceptance_probability(args,
user_value_preferences)
    graph = create_group_value_baf_graph(args)
    visualize_group_value_baf_graph(graph)


# Evaluates the value bipolar argumentation framework and shows its
graphical representation for an individual user.
def evaluate_individual_val_baf(args, user_value_preferences):
    calculate_individual_acceptance_probability(args,
user_value_preferences)
    graph = create_individual_value_baf_graph(args)
    visualize_individual_value_baf_graph(graph)


# Define the arguments
arguments = [
    Argument('*'),
    Argument('A'),
    Argument('B'),
    Argument('C'),
    Argument('D'),
    Argument('E'),
    Argument('F'),
    Argument('G'),
    Argument('H'),
    Argument('I'),
    Argument('J'),
    Argument('K'),
    Argument('L'),
    Argument('M'),
    Argument('N'),
    Argument('O'),
]
```

```
# Define the attacks
arguments[0].attacks = ['A', 'L']
arguments[1].attacks = ['B', 'C']
arguments[6].attacks = ['I']
arguments[10].attacks = ['K']
arguments[12].attacks = ['O']

# Define the supports
arguments[0].supports = ['E', 'F', 'J']
arguments[1].supports = ['D']
arguments[6].supports = ['H', 'G']
arguments[12].supports = ['M', 'N']

# Define the values for each argument. NB There is no need to define a
value for initial claim.
arguments[1].value = 'human wellbeing'
arguments[2].value = 'trust in experts'
arguments[3].value = 'personal autonomy'
arguments[4].value = 'human wellbeing'
arguments[5].value = 'knowledge'
arguments[6].value = 'knowledge'
arguments[7].value = 'knowledge'
arguments[8].value = 'knowledge'
arguments[9].value = 'knowledge'
arguments[10].value = 'knowledge'
arguments[11].value = 'human wellbeing'
arguments[12].value = 'knowledge'
arguments[13].value = 'knowledge'
arguments[14].value = 'human wellbeing'
arguments[15].value = 'knowledge'

# Define the value preference orderings for each user (the higher the
number the more preferred).
# NB the actual numbers used are not important just the relative ranking
for each user.
user_value_preferences = [
    {'*': 0, 'trust in experts': 1, 'knowledge': 3, 'human wellbeing': 2,
'personal autonomy': 4},
    {'*': 0, 'trust in experts': 1, 'knowledge': 2, 'human wellbeing': 4,
'personal autonomy': 3},
    {'*': 0, 'trust in experts': 1, 'knowledge': 2, 'human wellbeing': 4,
'personal autonomy': 3},
    {'*': 0, 'trust in experts': 3, 'knowledge': 4, 'human wellbeing': 2,
'personal autonomy': 1}
    ]


evaluate_individual_val_baf(arguments, user_value_preferences[0])
evaluate_group_val_baf(arguments, user_value_preferences)
```

Example Individual User Evaluation

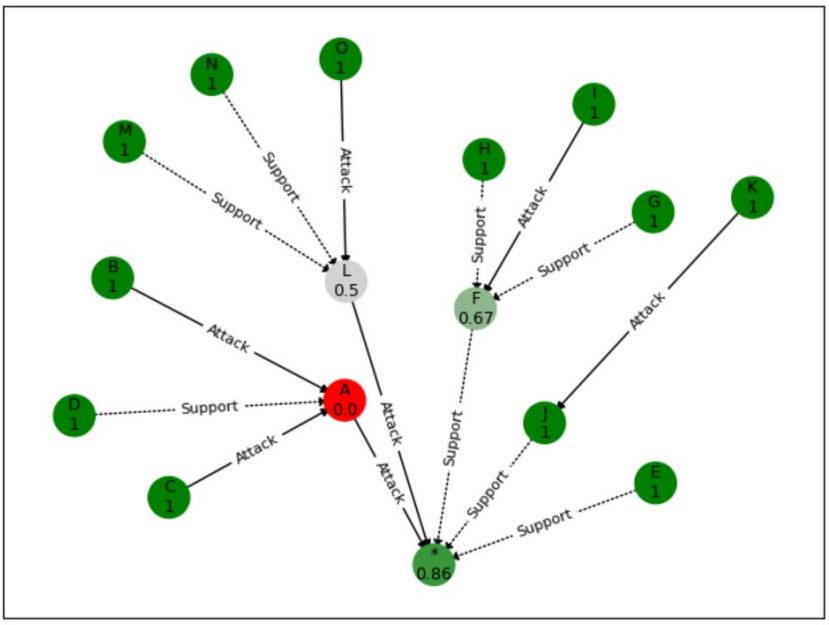


Figure 30: *An example evaluation output of the Value-Based System for an individual user.*

The output shows that the user is in favour of the initial claim (*) with an acceptance probability of 0.86 (ie greater than 0.5).

Example Group Evaluation

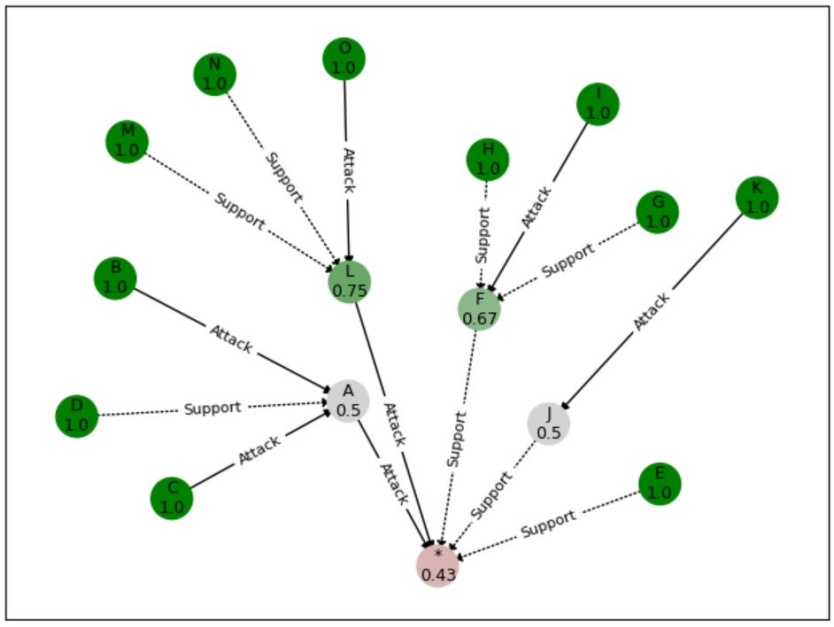


Figure 31: *An example evaluation output of the Value-Based System for a group.*

The output shows that the group is not in favour of the initial claim (*), with an average acceptance probability of 0.43 (ie less than 0.5).

D Voting-Based System Source Code

The following pages contain the source code for my Voting-Based System and an example evaluation using the argumentation framework from the test case in Appendix E.

The code has been written in Python 3.9, and makes use of the following open-source non-standard libraries:

- Matplotlib: *a comprehensive library for creating static, animated, and interactive visualizations in Python.* [29]
- NetworkX: *a package for the creation, manipulation, and study of the structure, dynamics, and functions of complex networks.* [32]

I verify that I am the sole author of the programmes contained in this appendix, except where explicitly stated to the contrary.

William Feeney. 11 August 2023.

```
# The code below implements my Voting-Based System. The argumentation
framework can be changed
# by changing the arguments, attacks and supports towards the end of the
code.  The voting input
# can also be changed and/or additional users added at the end of the code.

# NB the voting aggregation function is the function described in Section
4. If a different categories of
# votes or a different voting function is to be used then the
'calculate_vbs' function within the
# Argument class will need to be updated.

import networkx as nx
import matplotlib.pyplot as plt
from matplotlib.colors import LinearSegmentedColormap

# Creates a class for the arguments, containing information on their id
(typically a letter), the arguments
# attacking it, the arguments supporting it, its vote, its vote base score
and its final weight. NB the
# vote base score and final weight are initially set to 0.5, these are
updated when the function
# 'evaluate_vote_baf' is run.
class Argument:
    def __init__(self, id):
        self.id = id
        self.attacks = set()
        self.supports = set()
        self.votes = []
        self.vbs = 0.5
        self.weight = 0.5

    # Calculates the vote base score for the argument based on the input
user votes.
    def calculate_vbs(self):
        eps = 0.01
        if self.votes:
            weighted_score = 0
            for vote in self.votes:
                weighted_score += (vote - 1) * 0.2
            score = weighted_score / (len(self.votes) + eps)
            self.vbs = score
        else:
            self.vbs = 0.5

    # Calculates the final weight for the argument.
    def calculate_weight(self, args):
        if self.attacks == set() and self.supports == set():
            self.weight = self.vbs
            return self.weight
        else:
            E = 0
            for support in self.supports:
                for argument in args:
                    if support == argument.id:
                        E += argument.calculate_weight(args)
            for attack in self.attacks:
                for argument in args:
                    if attack == argument.id:
                        E -= argument.calculate_weight(args)
            return 1 - (1 - self.vbs**2) / (1 + self.vbs * (2 ** E))
```

```
# For a specified argument, returns the arguments that attack it.
def get_attacks(args):
    attacks = set()
    for argument in args:
        for attack in argument.attacks:
            attacks.add((attack, argument.id))
    return attacks


# For a specified argument, returns the arguments that support it.
def get_supports(args):
    supports = set()
    for argument in args:
        for support in argument.supports:
            supports.add((support, argument.id))
    return supports


# Calculates the vote base score for all arguments.
def calculate_all_vbs(args):
    for argument in args:
        argument.calculate_vbs()


# Calculates the final weight for all arguments.
def calculate_all_weights(args):
    for argument in args:
        argument.weight = argument.calculate_weight(args)


# Creates a graph representation of the voting bipolar argumentation
framework.
def create_vote_baf_graph(args):
    graph = nx.DiGraph()

    # Add arguments as nodes
    for argument in args:
        graph.add_node(argument.id, weight=argument.weight)

    # Add attacks as directed edges
    attacks = get_attacks(args)
    for attack in attacks:
        graph.add_edge(attack[0], attack[1], linestyle='solid')

    # Add supports as directed edges
    supports = get_supports(args)
    for support in supports:
        graph.add_edge(support[0], support[1], linestyle='dotted')

    return graph


# Creates a graphical representation of the vote bipolar argumentation
framework.
def visualize_vote_baf_graph(graph):
    pos = nx.spring_layout(graph, iterations=10000, seed=15)  # Seed can be
changed for other frameworks

    edge_styles = [graph[u][v]['linestyle'] for u, v in graph.edges()]
```

```
    node_weights = list(nx.get_node_attributes(graph, 'weight').values())
    node_labels = {node: f"{node}\n{round(graph.nodes[node]['weight'],2)}"
for node in graph.nodes()}
    edge_labels = {(u, v): "Attack" if graph[u][v]['linestyle'] == 'solid'
else "Support" for u, v in graph.edges()}

    plt.figure(figsize=(8, 6))

    # Creates a colour map for the nodes based on their final weight
    cmap = LinearSegmentedColormap.from_list(
        'custom_cmap',
        [(0, 'red'), (0.5, 'lightgrey'), (1, 'green')],
        N=256
    )

    # Adds the nodes, node labels, edges and edge labels
    nx.draw_networkx_nodes(graph, pos, node_color=node_weights, cmap=cmap,
vmin=0, vmax=1, node_size=500)
    nx.draw_networkx_labels(graph, pos, font_size=9, labels=node_labels)
    nx.draw_networkx_edges(graph, pos, edge_color='black',
style=edge_styles, arrows=True, arrowsize=10)
    nx.draw_networkx_edge_labels(graph, pos, edge_labels=edge_labels,
font_color='black', font_size=8)

    plt.axis('on')
    plt.show()


# Evaluates the voting bipolar argumentation framework and shows its
graphical representation
def evaluate_vote_baf(args):
    calculate_all_vbs(args)
    calculate_all_weights(args)
    bipolar_graph = create_vote_baf_graph(args)
    visualize_vote_baf_graph(bipolar_graph)


# Define the arguments
arguments = [
    Argument('*'),
    Argument('A'),
    Argument('B'),
    Argument('C'),
    Argument('D'),
    Argument('E'),
    Argument('F'),
    Argument('G'),
    Argument('H'),
    Argument('I'),
    Argument('J'),
    Argument('K'),
    Argument('L'),
    Argument('M'),
    Argument('N'),
    Argument('O'),
]

# Define the attacks
arguments[0].attacks = ['A', 'L']
arguments[1].attacks = ['B', 'C']
arguments[6].attacks = ['I']
```

```
arguments[10].attacks = ['K']
arguments[12].attacks = ['O']

# Define the supports
arguments[0].supports = ['E', 'F', 'J']
arguments[1].supports = ['D']
arguments[6].supports = ['H', 'G']
arguments[12].supports = ['M', 'N']

# Provide user voting input for each argument.  NB the system assumes the
same number of votes
# are provided for each argument.
arguments[1].votes = [2, 6, 5, 3]
arguments[2].votes = [6, 4, 4, 4]
arguments[3].votes = [5, 5, 2, 2]
arguments[4].votes = [2, 3, 5, 4]
arguments[5].votes = [3, 1, 1, 1]
arguments[6].votes = [1, 1, 2, 2]
arguments[7].votes = [1, 3, 2, 2]
arguments[8].votes = [2, 2, 3, 3]
arguments[9].votes = [4, 5, 5, 3]
arguments[10].votes = [4, 4, 3, 5]
arguments[11].votes = [6, 4, 6, 2]
arguments[12].votes = [1, 2, 4, 6]
arguments[13].votes = [2, 2, 5, 5]
arguments[14].votes = [5, 2, 4, 6]
arguments[15].votes = [2, 3, 5, 2]

evaluate_vote_baf(arguments)
```

Example Group Evaluation

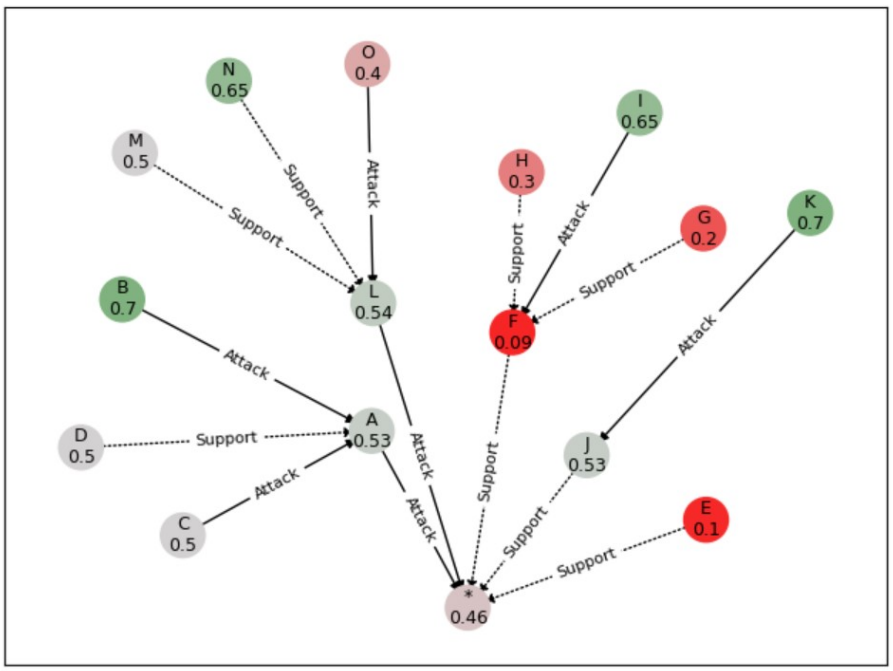


Figure 32: *An example evaluation output of the Voting-Based System for a group.*

The output shows that the group is not in favour of the initial claim (*), with a final weigthing of 0.46 (ie less than 0.5).

E User Testing Scenario, Questionnaire & Scenario Argumentation Framework

Scenario

You are a member of an advisory committee at the King's Space Agency, a private space company involved in not-for-profit space exploration and scientific research. The advisory committee is responsible for providing recommendations to the agency's administrator on various decisions related to mission planning and execution.

The agency is carrying out a mission to try to land the first humans on Mars using its rocket Ares. Ares is currently in orbit around Mars but has not yet attempted a landing due to prolonged stormy weather conditions on the surface of Mars, which increase the risk of potentially fatal accidents during landing. Ares has now been orbiting Mars for three weeks and, due to limited resources onboard on the rocket, this is now the last opportunity to attempt a landing. Otherwise, the mission will be aborted and Ares will return back to Earth without attempting to land. The crew of Ares have agreed unanimously if given permission they would be willing to attempt a landing.

As a member of the advisory committee, you have been asked to make a decision about whether or not to attempt a landing. To help your decision you have been presented with the main arguments for and against attempting a landing from other key agency staff members. Due to time restraints, you will be unable to get any further information. With the other members of the advisory committee you are required to consider these arguments and agree whether or not a landing should be attempted.

Key Arguments

A: "The landing process involves complex manoeuvres and high-speed descents, leaving limited margin for error. A single mishap during landing could result in catastrophic consequences, including the loss of the crew's lives. With the adverse weather conditions it is too risky to attempt a landing."

B: "In space travel there is no such thing as a risk-free landing. However, the crew are professional pilots with significant experience. They have practiced landing in similar conditions in a simulator, with a success rate of around 65-75% - compared to the success rate in average conditions of around 95% ."

C: "The crew have all privately said that for this mission they would be willing to take risks beyond what the agency would usually tolerate, even if there was an increased risk to their lives. As they have taken ownership of the possible consequences, they should be allowed to attempt a landing."

D: "The families of the crew need to be considered, especially the crew that have children. It is not fair on these families to take greater risks than were anticipated at the start of the

mission, and we do not have time to check if they would consent to the landing now."

E: "If we do not land now then it will be several years before another landing can be attempted, assuming we are even able to launch another attempt at all."

F: "For the first humans to set foot on Mars would be an extraordinary achievement in human history. It would represent a significant milestone in space exploration and a testament to human capabilities, ambition, and perseverance. Such an achievement would likely encourage further scientific progress."

G: "It would captivate the public's imagination and generate widespread excitement about space exploration, inspiring people of all ages and backgrounds, igniting interest in STEM fields. The event could stimulate educational initiatives, increase public support for space programs, and encourage the next generation of scientists, engineers, and astronauts."

H: "It would serve as a catalyst for international collaboration, fostering partnerships between nations and space agencies to further explore and develop Mars and beyond. The collaborative efforts and shared resources could accelerate progress in scientific research, technological advancements, and human colonization."

I: "Whilst landing on Mars would have symbolic value, the significance of the event is unlikely to translate into tangible benefits for society."

J: "Landing on Mars would provide astronauts with the ability to conduct hands-on research and exploration, which can yield more comprehensive and detailed results compared to remote observations. Direct access to the Martian surface would offer new insights into the planet's geology, climate, and potential for habitability."

K: "Robotic missions have already proven successful in exploring Mars and gathering valuable scientific data. Much of the research planned for this mission could be carried out at a later time by robots without the need for humans to be on the surface."

L: "If the landing goes wrong this may jeopardise the ability for future missions to go ahead, which could impact the wider community and not just King's Space Agency."

M: "Funding agencies, policymakers, and the public may be more hesitant to support future missions if there is a perception that the risk of failure is too high."

N: "Mission failures can have psychological effects on the individuals involved in the mission and the broader space community. This can lead to a loss of confidence, decreased morale, and a need for psychological recovery."

O: "Everyone is aware of the risks associated with space travel, and unfortunately that accidents will happen. The greater risk is if that we don't land then the funding agencies and public may lose confidence in our capabilities or lose interest in space exploration more generally."

Social Values Promoted by Each Argument [not seen by the participants]

- (a) Societal advancement of knowledge and exploration: E, F, G, H, I, J, L, M, O
- (b) Human welfare including preservation of life: A, D, K, N
- (c) Personal autonomy / Freedom of choice: C
- (d) Trust in the skills and competency of experts: B

Questionnaire

1. Please rank the following social values from the least important to the most important:
   (a) Societal advancement of knowledge and exploration
   (b) Human welfare including preservation of life
   (c) Personal autonomy / Freedom of choice
   (d) Trust in the skills and competency of experts
2. Please score every argument between 1 – 6 inclusive depending on your perceived strength of the argument in the context of the decision being made. 6 represents an argument you believe to be very strong, and 1 represents an argument you believe to be very weak. You can give multiple arguments the same score.
3. Please order the following groups of arguments from your least preferred argument (lowest scoring) to your most preferred argument (highest scoring). Where scores are tied you will still need to determine which argument is more preferred:
   (a) A, E, F, J, L
   (b) A, B, C, D
   (c) F, G, H, I
   (d) K, J
   (e) L, M, N, O
4. If you alone were required to make the decision about whether to attempt the landing or not, what would you do?

Scenario Argumentation Framework

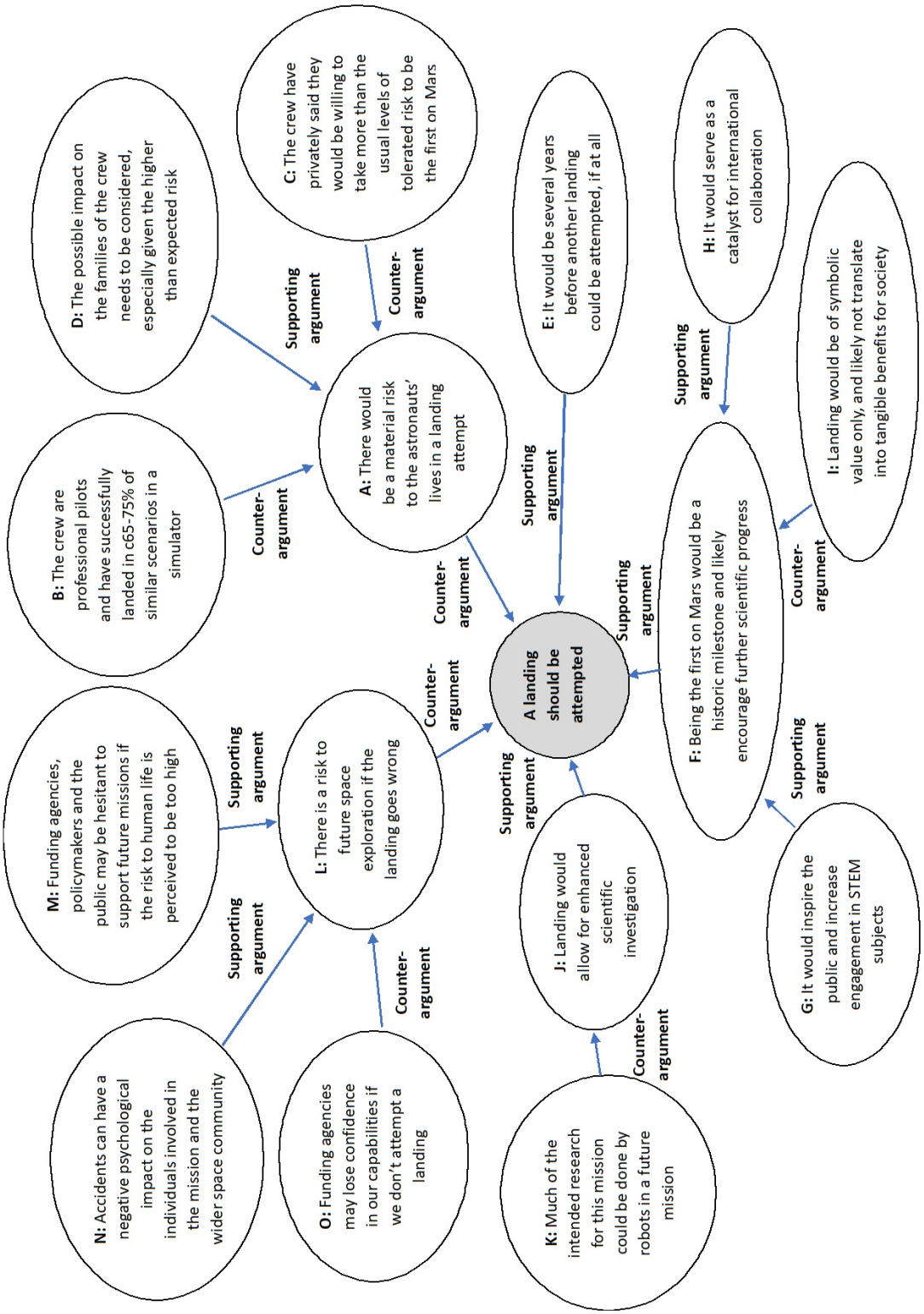


Figure 33: *Summary of the arguments provided to the participants.*

F User Testing Results

F.1 User Test 1

F.1.1 User Input: Value-Based System

User 1: c >> a >> b >> d

User 2: b >> c >> a >> d

User 3: b >> c >> a >> d

User 4: b >> c >> a >> d

F.1.2 Value-Based System Evaluation

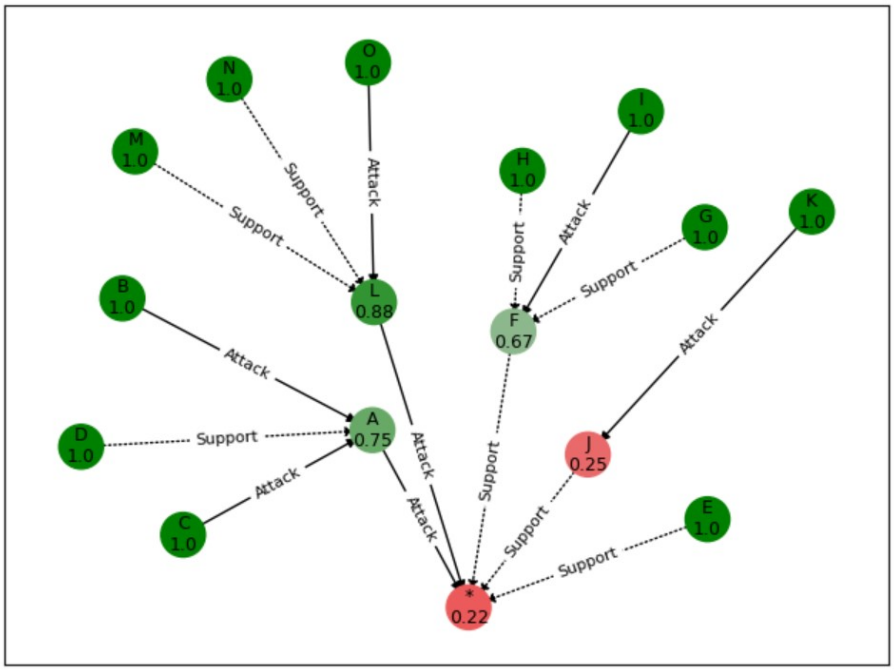


Figure 34: *Evaluation of Value-BAF for Group 1.*

F.1.3 User Input: Preference-Based System

User 1:
J >> E >> A >> F >> L
B >> C >> D >> A
I >> H >> G >> F
K >> J
N >> M >> O >> L

User 2:
A >> J >> L >> F >> E
A >> C >> B >> D

I >> G >> H >> F
K >> J
O >> N >> L >> M

User 3:
A >> L >> J >> F >> E
A >> D >> B >> C
I >> H >> G >> F
K >> J
M >> O >> L >> N

User 4:
L >> J >> A >> F >> E
B >> D >> A >> C
F >> G >> I >> H
J >> K
L >> N >> M >> O

F.1.4 Preference-Based System Evaluation

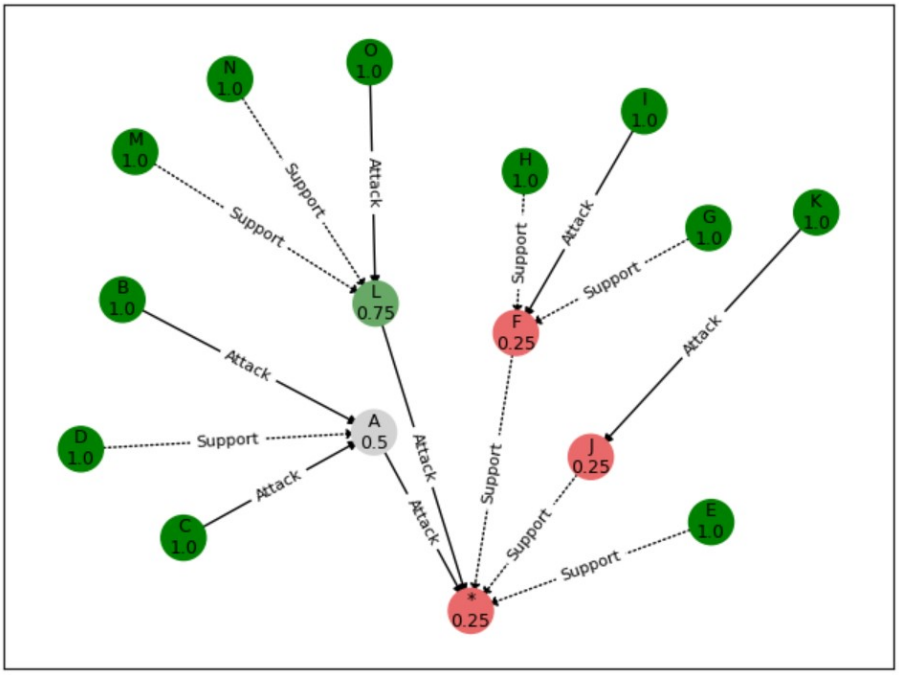


Figure 35: *Evaluation of Pref-BAF for Group 1.*

F.1.5 User Input: Voting-Based System

User 1: A-2, B-6, C-5, D-2, E-3, F-1, G-1, H-2, I-4, J-4, K-6, L-1, M-2, N-5, O-2

User 2: A-6, B-4, C-5, D-3, E-1, F-1, G-3, H-2, I-5, J-4, K-4, L-2, M-2, N-2, O-3

User 3: A-5, B-4, C-2, D-5, E-1, F-2, G-2, H-3, I-5, J-3, K-6, L-4, M-5, N-4, O-5

User 4: A-3, B-4, C-2, D-4, E-1, F-2, G-2, H-3, I-3, J-5, K-2, L-6, M-5, N-6, O-2

F.1.6 Voting-Based System Evaluation

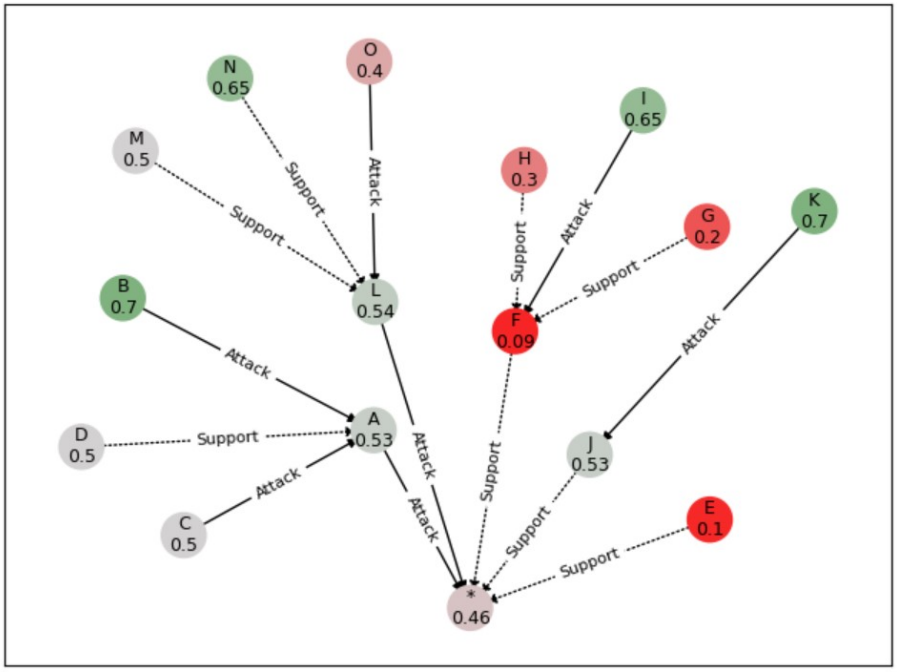


Figure 36: *Evaluation of Vote-BAF for Group 1.*

F.2 User Test 2

F.2.1 User Input: Value-Based System

User 5: a >> d >> c >> b

User 6: c >> b >> d >> a

User 7: a >> b >> d >> c

F.2.2 Value-Based System Evaluation

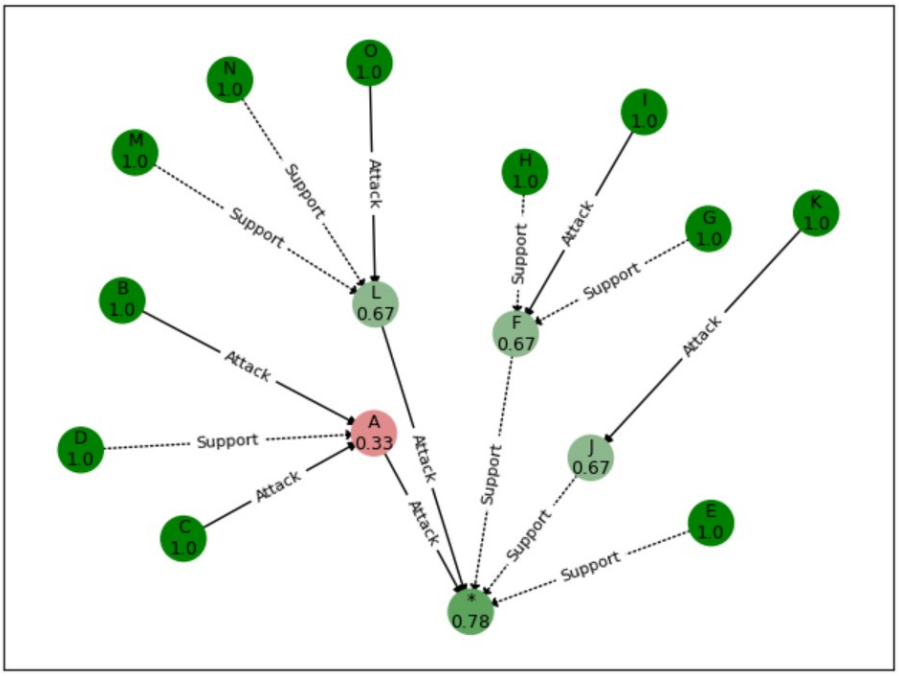


Figure 37: *Evaluation of Value-BAF for Group 2.*

F.2.3 User Input: Preference-Based System

User 5:
J >> F >> L >> A >> E
B >> C >> D >> A
F >> I >> G >> H
J >> K
L >> M >> N >> O

User 6:
A >> F >> J >> E >> L
A >> B >> C >> D
F >> H >> G >> I
K >> J
N >> O >> M >> L

User 7:
L >> J >> F >> E >> A
B >> C >> A >> D
H >> G >> F >> I
J >> K
M >> O >> L >> N

F.2.4 Preference-Based System Evaluation

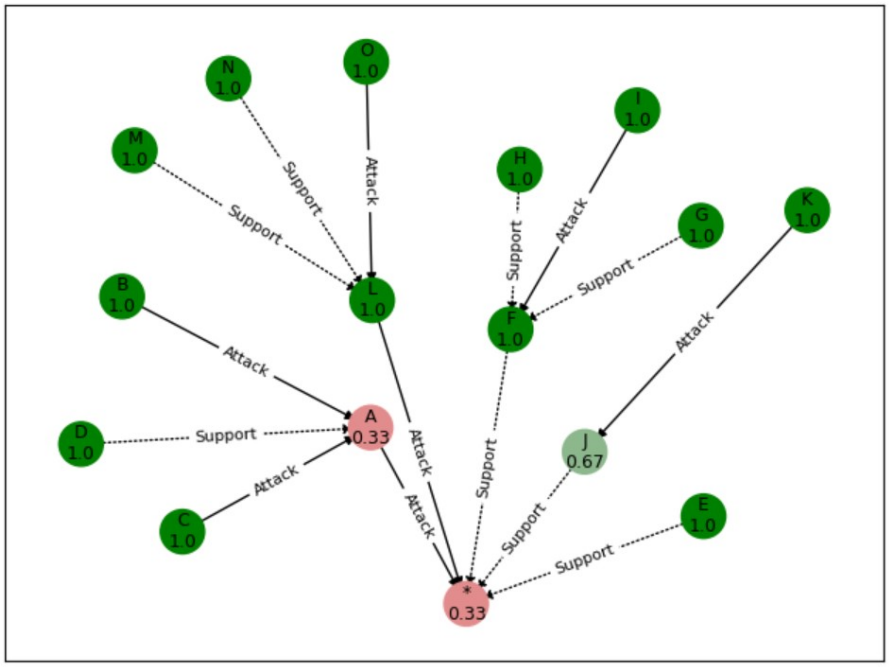


Figure 38: *Evaluation of Pref-BAF for Group 2.*

F.2.5 User Input: Voting-Based System

User 5: A-2, B-6, C-6, D-3, E-1, F-5, G-3, H-2, I-5, J-6, K-2, L-4, M-2, N-2, O-1

User 6: A-5, B-4, C-3, D-2, E-1, F-3, G-2, H-2, I-1, J-2, K-4, L-1, M-1, N-2, O-1

User 7: A-1, B-6, C-4, D-1, E-3, F-4, G-4, H-4, I-2, J-5, K-2, L-5, M-5, N-2, O-5

F.2.6 Voting-Based System Evaluation

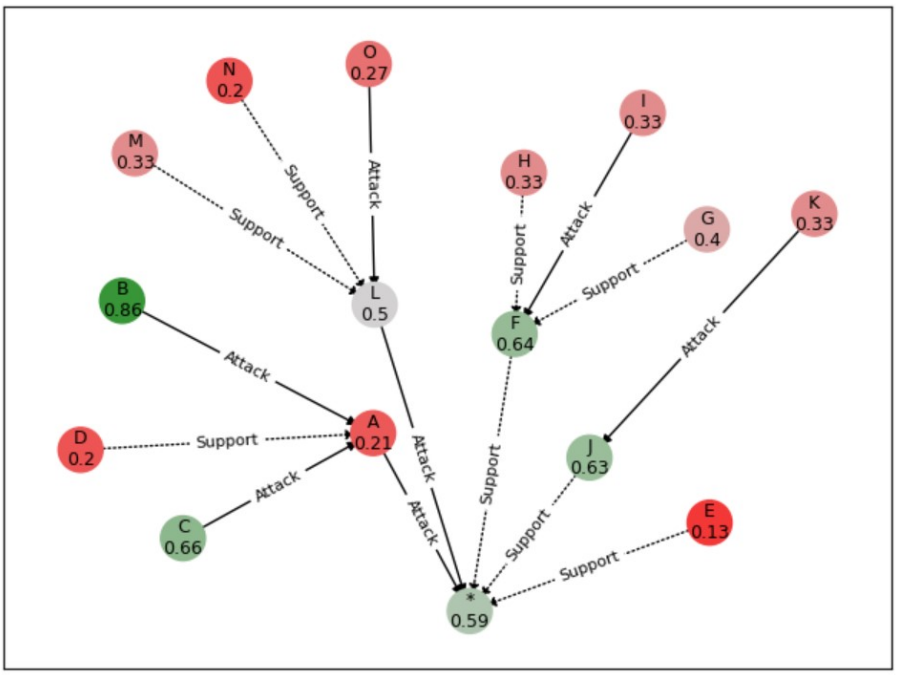


Figure 39: *Evaluation of Vote-BAF for Group 2.*

Bibliography

References

[1] BCS (2023). https://www.bcs.org/membership-and-registrations/become-a-member/bcs-code-of-conduct/. BCS Code of Conduct. Accessed: 05-07-2023.

[2] Teresa Alsinet, Josep Argelich, Ramón Béjar, Cèsar Fernández, Carles Mateu, and Jordi Planes. Weighted argumentation for analysis of discussions in twitter. *International Journal of Approximate Reasoning*, 85:21–35, 2017.

[3] Leila Amgoud and Jonathan Ben-Naim. Evaluation of arguments in weighted bipolar graphs. *International Journal of Approximate Reasoning*, 99:39–55, 2018.

[4] Leila Amgoud and Claudette Cayrol. A reasoning model based on the production of acceptable arguments. *Annals of Mathematics and Artificial Intelligence*, 34:197–215, 2002.

[5] Leila Amgoud, Claudette Cayrol, Marie-Christine Lagasquie-Schiex, and Pierre Livet. On bipolarity in argumentation frameworks. *International Journal of Intelligent Systems*, 23(10):1062–1093, 2008.

[6] Kenneth J Arrow. A difficulty in the concept of social welfare. *Journal of political economy*, 58(4):328–346, 1950.

[7] Pietro Baroni, Marco Romano, Francesca Toni, Marco Aurisicchio, and Giorgio Bertanza. Automatic evaluation of design alternatives with quantitative argumentation. *Argument & Computation*, 6(1):24–49, 2015.

[8] Trevor JM Bench-Capon. Persuasion in practical argument using value-based argumentation frameworks. *Journal of Logic and Computation*, 13(3):429–448, 2003.

[9] Trevor JM Bench-Capon and Paul E Dunne. Argumentation in artificial intelligence. *Artificial intelligence*, 171(10-15):619–641, 2007.

[10] Duncan Black. On the rationale of group decision-making. *Journal of political economy*, 56(1):23–34, 1948.

[11] Dries H Bostyn, Sybren Sevenhant, and Arne Roets. Of mice, men, and trolleys: Hypothetical judgment versus real-life behavior in trolley-style moral dilemmas. *Psychological science*, 29(7):1084–1093, 2018.

[12] Martin Caminada. A gentle introduction to argumentation semantics. *Lecture material, Summer*, 2008.

[13] Paulius Čerka, Jurgita Grigienė, and Gintarė Sirbikytė. Is it possible to grant legal personality to artificial intelligence software systems? *Computer law & security review*, 33(5):685–699, 2017.

[14] Carlos Iván Chesnevar, Ana Gabriela Maguitman, and Ronald Prescott Loui. Logical models of argument. *ACM Computing Surveys (CSUR)*, 32(4):337–383, 2000.

[15] Simon Chesterman. Artificial intelligence and the limits of legal personality. *International & Comparative Law Quarterly*, 69(4):819–844, 2020.

[16] Shai Danziger, Jonathan Levav, and Liora Avnaim-Pesso. Extraneous factors in judicial decisions. *Proceedings of the National Academy of Sciences*, 108(17):6889–6892, 2011.

[17] Phan Minh Dung. On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games. *Artificial intelligence*, 77(2):321–357, 1995.

[18] Paul E Dunne, Anthony Hunter, Peter McBurney, Simon Parsons, and Michael Wooldridge. Weighted argument systems: Basic definitions, algorithms, and complexity results. *Artificial Intelligence*, 175(2):457–486, 2011.

[19] Sinan Eğilmez, Joao Martins, and Joao Leite. Extending social abstract argumentation with votes on attacks. In *Theory and Applications of Formal Argumentation: Second International Workshop, TAFA 2013, Beijing, China, August 3-5, 2013, Revised Selected papers 2*, pages 16–31. Springer, 2014.

[20] Oriel FeldmanHall, Dean Mobbs, Davy Evans, Lucy Hiscox, Lauren Navrady, and Tim Dalgleish. What we say and what we do: The relationship between real and hypothetical moral choices. *Cognition*, 123(3):434–441, 2012.

[21] Nick Fletcher MP [@NickFletcherMP]. [Twitter] 9 February 2023. Available at: https://twitter.com/NickFletcherMP/status/1623699476366991360. Accessed on 28 February 2023.

[22] Department for Science Innovation and Technology (2023). `https://www.gov.uk/government/publications/ai-regulation-a-pro-innovation-approach/white-paper`. A pro-innovation approach to AI regulation (command paper number: 815). Accessed: 23-06-2023.

[23] Allan Gibbard. Manipulation of voting schemes: a general result. *Econometrica: journal of the Econometric Society*, pages 587–601, 1973.

[24] Maralee Harrell. Understanding, evaluating, and producing arguments: Training is necessary for reasoning skills. *Behavioral and Brain Sciences*, 34(2):80, 2011.

[25] Daniel Kahneman. *Thinking, fast and slow*. macmillan, 2011.

[26] George Kateb. *Human dignity*. Harvard University Press, 2014.

[27] Arto Laitinen and Otto Sahlgren. Ai systems and respect for human autonomy. *Frontiers in artificial intelligence*, 4:705164, 2021.

[28] Joao Leite and Joao Martins. Social abstract argumentation. In *Twenty-Second International Joint Conference on Artificial Intelligence*, 2011.

[29] Matplotlib. https:matplotlib.org/. Accessed: 22-06-2023.

[30] Sanjay Modgil. Reasoning about preferences in argumentation frameworks. *Artificial intelligence*, 173(9-10):901–934, 2009.

[31] Tony Moore. "Australian Democrats pledge to 'keep the bastards honest' one more time". Brisbane Times. [Online]. 6 October 2021. Retrieved from URL: https://www.brisbanetimes.com.au/politics/federal/australian-democrats-pledge-to-keep-the-bastards-honest-one-more-time-20211005-p58xe9.html. Accessed on 8 March 2023.

[32] NetworkX. https://networkx.org/. Accessed: 22-06-2023.

[33] Hung T Nguyen, Olga Kosheleva, and Vladik Kreinovich. So how to make group decisions? arrow's impossibility theorem 70 years after. *Asian Journal of Economics and Banking*, 5(3):226–233, 2021.

[34] Future of Life Institute (2017). https://futureoflife.org/open-letter/ai-principles/. AI Principles. Accessed: 25-06-2023.

[35] Regulation (EU) 2016/679 of the European Parliament, of the Council of 27 April 2016 on the protection of natural persons with regard to the processing of personal data, and on the free movement of such data (United Kingdom General Data Protection Regulation) (Text with EEA relevance) (2016). https://www.legislation.gov.uk/eur/2016/679. Accessed: 23-06-2023.

[36] H. G. Plato. *The Collected Dialogues of Plato*. Princeton University Press, 1961.

[37] John L Pollock. Defeasible reasoning. *Cognitive science*, 11(4):481–518, 1987.

[38] John L Pollock. A theory of defeasible reasoning. *International Journal of Intelligent Systems*, 6(1):33–54, 1991.

[39] John L Pollock. Oscar: An agent architecture based on defeasible reasoning. In *AAAI spring symposium: emotion, personality, and social behavior*, pages 55–60. Citeseer, 2008.

[40] W Nicholson Price, Sara Gerke, and I Glenn Cohen. Potential liability for physicians using artificial intelligence. *Jama*, 322(18):1765–1766, 2019.

[41] Antonio Rago and Francesca Toni. Quantitative argumentation debates with votes for opinion polling. In *PRIMA 2017: Principles and Practice of Multi-Agent Systems: 20th International Conference, Nice, France, October 30–November 3, 2017, Proceedings 20*, pages 369–385. Springer, 2017.

[42] Antonio Rago, Francesca Toni, Marco Aurisicchio, Pietro Baroni, et al. Discontinuity-free decision support with quantitative argumentation debates. *KR*, 16:63–73, 2016.

[43] Raymond Reiter. A logic for default reasoning. *Artificial intelligence*, 13(1-2):81–132, 1980.

[44] John Rothchild. Introduction to athenian democracy of the fifth and fourth centuries bce. *Wayne State University Law School Research Paper*, (07-32), 2007.

[45] Richard M Ryan. *Self-Determination Theory: Basic Psychological Needs in Motivation, Development, and Wellness*. The Guilford Press A Division of Guilford Publications, Inc. New York, 2017.

[46] Mark Allen Satterthwaite. Strategy-proofness and arrow's conditions: Existence and correspondence theorems for voting procedures and social welfare functions. *Journal of economic theory*, 10(2):187–217, 1975.

[47] Lily Ross Taylor. *Roman voting assemblies: from the Hannibalic War to the dictatorship of Caesar*, volume 8. University of Michigan Press, 1990.

[48] Marlene E Turner and Anthony R Pratkanis. Twenty-five years of groupthink theory and research: Lessons from the evaluation of a theory. *Organizational behavior and human decision processes*, 73(2-3):105–115, 1998.

[49] Alexandros Vassiliades, Nick Bassiliades, and Theodore Patkos. Argumentation and explainable artificial intelligence: a survey. *The Knowledge Engineering Review*, 36:e5, 2021.

[50] Gerard AW Vreeswijk. Abstract argumentation systems. *Artificial intelligence*, 90(1-2):225–279, 1997.

[51] Douglas Walton. Argumentation theory: A very short introduction. In *Argumentation in artificial intelligence*, pages 1–22. Springer, 2009.

[52] Douglas Walton. *Argumentation schemes for presumptive reasoning*. Routledge, 2013.