Imperial College London

Interim Report

Web-Based Argumentation

Author:
Michael Procopiou

Supervisors:
Dr. Xiuyi Fan
Prof. Francesca Toni

January 31, 2014

Contents

1 Introduction

Argumentation is increasingly becoming a popular aspect of Artificial Intelligence that deals with the creation of knowledge systems and evaluation of decision problems. Past and current research has allowed for the development of various argumentation frameworks that allow the analysis of arguments by formulating them in a formal manner and evaluating them. This project focuses mostly on Assumption-Based Argumentation (ABA) and its implementation through a practical argumentation system. ABA is a form of argumentation where arguments and attacks are notions derived from primitive notions of rules in a deductive system, assumptions and contraries thereof [9]. This framework allows for the evaluation of a conclusion based on whether it is supported by a "winning" set of arguments. Such sets of arguments and assumptions can be determined as "winning/acceptable" based on a number of different semantics that ABA supports, which are discussed in more detail later on in the report.

Therefore, ABA is a potentially powerful tool which one could use to establish the validity of the conclusion put forward. Early successful application of argumentation theory and the ABA framework have occurred in fields such as legal-reasoning, medical diagnosis and decision theory. Within these fields ABA has been used not only because of its ability to determine propositions as acceptable, but also due to its ability to convey the derivation process to the user. There are various computational mechanisms that have been devised to algorithmically compute the acceptability of a claim. Using these mechanisms argumentation engines, such as proxdd [8] and grapharg [3], have been implemented thus allowing for the computation of acceptability of a claim under a defined ABA framework. Nonetheless, these engines are still at a primitive stage and require enhancing for ABA to become a widely accepted and applied tool in the real world.

Namely, there is a lack of an application that is easily accessible to users and provides both satisfactory performance and useful visualisations of the argument. The difficulties involved with creating such an application are multifold. These involve the performance of the underlying engine when dealing with large scale real problems, the difficulty a user might face in devising a correct ABA framework for their problem and devising a visualisation system to accurately convey the derivation process to the user. Additionally, the engines themselves are still at an early stage and need to be extended. For example the proxdd and grapharg systems lack the ability to derive the acceptability of a claim based on "ideal" semantics. Lastly, there is a lack of generalisation of these systems. At their current state they aim at simply resolving ABA frameworks directly implemented into the engine. This re-

stricts the usability of the system to users who intend to use just ABA and have significant knowledge of ABA to device the initial framework of their problem.

The project's objectives centre around providing solutions to the problems mentioned above by fulfilling the requirement for a web-based argumentation application that will allow users to exploit the existing ABA systems. The web-application should act as a platform that will provide the users with good user experience and the ability to easily and as seamlessly as possible interact with the ABA systems in the back end. Having computed the necessary derivation the user will be provided with the output in a useful and meaningful manner in the form of a visualisation. The end system would be similar to the ASPARTIX system implemented by TU Wien [6]. Having established the web-application, the project will then seek to enhance the existing systems by enabling them to compute based on more semantics mentioned above. Potentially the project might look in the development of a simple API for these engines. Lastly, the web-application might be extended to support Abstract Argumentation through its mapping to ABA (as described by [7]) and allow the direct input of decision problems (as described in [11]).

2 Background

2.1 Argumentation

Argumentation in general involves studying how claims and conclusions can be reached by applying logical reasoning. It is useful when analysing arguments in the form of dialogue and persuasion, especially under the context of philosophy, medicine and law. Research in artificial intelligence is looking to exploit the underlying logic in analysing arguments to allow for this analysis to be carried out in a computerised manner. In order to do so general purpose argumentation frameworks have been devised to allow for analysis of an argument as a computerised argumentation problem. Two such argumentation frameworks are Abstract Argumentation (AA) and Assumption-Based Argumentation (ABA). Argumentation frameworks such as these can provide the general basis required to implement specific frameworks for real-life problems where argumentation could help in decision making and argument analysis. For the web-based application proposed by this project the underlying framework will be ABA.

2.2 Assumption-Based Argumentation

Although ABA is an instance of AA, there are core differences in the way ABA perceives and evaluates arguments. In ABA arguments and attacks are derived from given rules in a deductive system, assumptions and their contraries [9]. An argument is supported by assumptions and is attacked by other arguments when the contrary of an assumption of the initial argument can be deduced by the opposing argument. However, in order for an argument to be deemed "acceptable" or "winning", ABA is equipped with numerous semantics that allow us to determine an "acceptable/winning" set of assumptions, from which we can deduce an "acceptable" set of arguments. Additionally several computational mechanism have been developed for ABA (see [8] and [3]) that allow a conclusion or claim to be algorithmically evaluate in terms of a "acceptable" set of arguments. The computational nature of these mechanisms allow us to programmatically implement the semantics and allow for computerised argumentation analysis of a framework.

2.3 The ABA framework

The potential power of ABA lies with its ability to represent and evaluate real-world decision problems. An implementation of ABA could prove to be an invaluable tool when it comes to evaluating potential claims, providing

support to existing claims or disputing existing claims, in any context as long as it can be generalised and implemented as part of the ABA framework.

The following is an example taken from [9] to demonstrate how ABA can interpret a simple real-world problem, while introducing the ABA framework and its elements.

"You like eating food, this makes you really happy. But you are very health-conscious too, and don't want to eat without a fork and with dirty hands. You are having a walk with some friends, and your friend Nelly is offering you all a piece of lasagne, looking really delicious. You are not quite sure, but typically you carry no cutlery when you go for walks, and your hands will almost certainly be dirty even if the may not look so. Should you go for the lasagne, trying to snatch it quickly from the other friend? You may argue with yourself as follows:

α : let me go for the lasagne, it looks delicious, and eating it will make me happy!

β : but I am likely to have no fork and my hands will be dirty, I don't want to eat it in these circumstances, let the others have it!

α and β can be seen as arguments, and β disagrees with (attacks) α. If these are all the arguments put forward, since no argument is proposed that defends α against β , the decision to go for the lasagne is not dialectically justified. If, however, you put forward the additional argument

γ : who cares about the others, let me get the lasagne, I may have a fork after all

This provides a defence for α against β, and the decision to go for the lasagna becomes dialectically (although possibly not ethically) justified."

This example is illustrative of the breadth of problems that could potentially be analysed by an ABA system. Granted the example is overly simplistic. It has a limited amount of arguments and accounts for just a few of the possible parameters that could affect whether we are happy or not. Nonetheless, being a general framework the simplicity of the problem does not affect our ability to device a specific framework for this problem from which we can evaluate our claim of being happy (or not being happy).

The ABA frame work is defined by [9] as follows:

Definition 1. An ABA framework is a tuple $\langle \mathcal{L}, \mathcal{R}, \mathcal{A}, \ ^- \ \rangle$ where

- $\langle \mathcal{L}, \mathcal{R} \rangle$ is a deductive system, with $\mathcal{L}$ the *language* and $\mathcal{R}$ a set of *rules*, that we assume of the form $\sigma_0 \leftarrow \sigma_1, \ldots, \sigma_m (m \geq 0)$ with $\sigma_i \in \mathcal{L} (i = 0, \ldots, m)$; σ_0 is referred to as the *head* and $\sigma_i, \ldots, \sigma_m$ as the *body* of the rule $\sigma_0 \leftarrow \sigma_1, \ldots, \sigma_m$;
- $\mathcal{A} \subseteq \mathcal{L}$ is a (non-empty) set, referred to as *assumptions*;

- $^{-}$ is a total mapping from $\mathcal{A}$ into $\mathcal{L}$; $\bar{\alpha}$ is referred to as the *contrary* of α.

Referring back to the example used let us assume that the following mapping exists:

- p - *happy*
- r - *not happy*
- a - *eating*
- q - *good food*
- b - *no fork*
- c - *dirty hands*
- s - *fork*
- t - *clean hands*

Based on this mapping and the definition of the an ABA framework we can now construct a framework that represents the example, as shown in example 1:

Example 1. $\langle \mathcal{L}, \mathcal{R}, \mathcal{A}, ^{-} \rangle$ may consist of
$\mathcal{L} = \{a, b, c, p, q, r, s, t\}$
$\mathcal{R} = \{p \leftarrow q, a, q \leftarrow, r \leftarrow b, c\}$
$\mathcal{A} = \{a, b, c\}$
$\bar{a} = r, \bar{b} = s, \bar{c} = t$

The framework constructed in example 1 displays the various elements that allow one to represent a real-life decision problem in ABA. It can be seen that the rules related to our problem are defined in terms of inference rules. Additionally, the framework also includes the following elements:

Assumptions are, in our example, actions that can be taken or disputable observations about the argument. In general assumptions can be considered as vulnerable points of an argument that can be used to support or dispute that argument. The aim of ABA is to establish a set of "strong" assumptions that support or dispute the underlying argument.

Non-Assumptions represent goals we wish to achieve or avoid in accordance to the claim we are trying to analyse. These can be based on certain assumption we are considering or might be stand-alone goals.

Contraries are sentences that are opposing to the assumptions of our framework. ABA imposes that each assumption must be mapped to a contrary. For example, the assumption "no fork" in our framework has a contrary of "fork". Thus our framework is now equipped with both the assumptions necessary but also the opposing assumptions that can dispute them.

The description provides a simple overview of what these elements of the framework are and are used in the example. This will allow us to explore how claims are analysed. For the purpose of this report there is no need to explore to a deeper level the framework, however there are numerous publications that explain in these elements in more detail (see [9]).

2.4 Proving a claim using ABA

The ability of ABA to evaluate the support of a claim, lies in its ability to formulate and analyse arguments. Arguments are the deduction of a certain claim from the rules and assumptions provided. The following is a definition of an argument (according to [9]):

Definition 2. an argument is defined as follows:
An argument for (the claim) $\sigma \in \mathcal{L}$ supported by $A \subseteq \mathcal{A}$ ($A \vdash \sigma$ in short) is a deduction for σ supported by A (and some R $\subseteq \mathcal{R}$)

However, within the ABA framework there is also the notion of attack (similarly to AA). An attack between arguments under ABA occurs when one argument deduces the contrary of an assumption of another argument. Attacks can be defined as follows (from [9]):

Definition 3. an attack is defined as follows:
An argument $A_1 \vdash \sigma_1$ attacks an argument $A_2 \vdash \sigma_2$ iff σ_1 is the contrary of one of the assumptions in A_2.

ABA is equipped with various semantics that allow it to determine "winning/acceptable" sets of arguments and assumptions. This enables us to validate whether a claim can be deduced from these "acceptable" arguments and whether the claim is supported by these "winning" assumptions. These semantics can define the "acceptability" of both a set of arguments and a set of assumptions. Both approaches (or views) are defined below (in accordance to [9]):

Argument View Semantics

- *admissible* iff it does not attack itself and it attacks all arguments that attack it;
- *preferred* iff it is maximally (w.r.t. $\subseteq$) admissible;

- *sceptically preferred* iff it is the intersection of all preferred sets of arguments;
- *complete* iff it is admissible and contains all arguments it *defends*, where A defends α iff A attacks all arguments that attack α;
- *grounded* iff it is minimally (w.r.t $\subseteq$) complete;
- *ideal* iff it is maximally (w.r.t $\subseteq$) admissible *and* contained in all preferred sets of arguments;
- *stable* iff it does not attack itself and it attacks all arguments it does not contain.

Assumption View Semantics

- *admissible* iff it does not attack itself and it attacks all assumptions that attack it;
- *preferred* iff it is maximally (w.r.t. $\subseteq$) admissible;
- *sceptically preferred* iff it is the intersection of all preferred sets of assumptions;
- *complete* iff it is admissible and contains all assumptions it *defends*, where A defends α iff A attacks all assumptions that attack α;
- *grounded* iff it is minimally (w.r.t $\subseteq$) complete;
- *ideal* iff it is maximally (w.r.t $\subseteq$) admissible *and* contained in all preferred sets of assumptions;
- *stable* iff it does not attack itself and it attacks all assumptions it does not contain.

2.5 Computational mechanisms for ABA

The power of ABA is unleashed through the various computational mechanisms and tools that have been developed. These allow for ABA frameworks to be analysed in an algorithmic manner, thus allowing the creation of computerised systems that can carry out this analysis. The underlying features of these mechanisms are as follows (as described by [9]):

- they can be abstracted away as disputes between a *proponent* and an *opponent*, in a sort of (fictional) zero-sum, two-player game: the proponent aims at proving (constructively) that an initially given *sentence* is "acceptable"/"winning", the opponent is trying to prevent the proponent from doing so;
- these disputes are defined as a *sequence of tuples* (referred to as *dispute derivations*), representing the state of the game while it is being played;

- these disputes interleave the construction of arguments and identification of attacks between them with testing whether the input sentence is "acceptable"/"winning";
- the rules of the game allow proponent and opponent to perform various kinds of *filtering* during disputes, but different kinds for different argumentation semantics (for determining whether the input sentence is "acceptable"/"winning";
- the possible outcomes of dispute derivations are as follows:
  - the input sentence is proven to be "acceptable"/"winning" (the dispute derivation is *successful*) or is not proven to be so; and
  - if the dispute derivation is successful, it returns:
    1. the set of all assumptions supporting the arguments by the proponent(referred to as the *defence set*),
    2. the set of all assumptions in the support of arguments by the opponent and chosen by the proponent to be counter-attacked (referred to as the *culprits*),
    3. in the case of the proposal of ([8]), the *dialectical tree* of arguments by proponent and opponent.

With these features in mind several procedure and algorithms have been devised that determine whether a claim is "acceptable/winning" based on several of the semantics mentioned. These include a wide range of tools from flowcharts to logic programming systems, some of which are discussed in the following sections.

2.6 Proxdd

Proxdd (refer to [2]) is a system developed by Dr. Robert Craven that implements dispute derivations to analyse ABA frameworks. Given a constructed framework it can algorithmically analyse and return a set of winning arguments in accordance to the semantics specified. Proxdd currently supports admissible and grounded semantics. It is implemented using the Prolog logic programming language (for more details see [2]). The underlying computational mechanism used for Proxdd is the SXDD algorithm (defined in [8]).

Example 2. Example of input (from [2])

```
myAsm(a).
myAsm(b).
myAsm(c).
myAsm(d).
```

```
myAsm(e).
myAsm(f).
contrary(a, q).
contrary(b, f).
contrary(c, u).
contrary(d, v).
contrary(e, v).
contrary(f, v).
myRule(p, [a,u]).
myRule(q, [b,r]).
myRule(q, [c,s]).
myRule(q, [c,t]).
myRule(u, [a]).
myRule(s, []).
myRule(t, [d]).
myRule(t, [e]).
```

Example 3. Example of output (from [2])

```
DEF: [a]
CUL: [c]
ARG: [1:([a], [] -> p),6:([c], [] -> q),7:([a], [] -> u),
 8:([c], [d] -> q),9:([c], [e] -> q)]
ATT: [1-0,6-1,7-6,8-1,9-1]
```

2.7 Grapharg

Grapharg (refer to [1]) is a system also developed by Dr. Robert Craven. Similarly to Proxdd it also provides dispute derivation for ABA. Furthermore, it is also implemented in Prolog, however the underlying algorithm is different. Unlike Proxdd it uses a graph-based dispute derivation algorithm that, according to the research's findings, optimises the computation of the dispute derivation (refer to [3]).

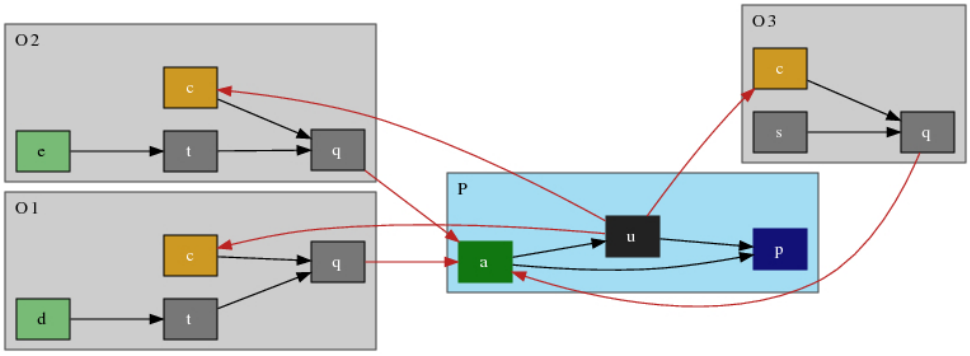
In both Proxdd and Grapharg the system takes in a framework in the form shown in example 2 and after the dispute derivation it produces an output similar to example 4. Additionally, by using graphviz, the output can be displayed graphically as in example 5.

Example 4. Example of output (from [1])

```
DEFENCE: [a]
```

```
CULPRITS: [c]
PROP JUSTIFICATIONS: [(a,*),(p,1),(u,5)]
OPP JUSTIFICATIONS: [[]-[(c,*),(d,*),(q,4),(t,7)]-[q],
[]-[(c,*),(e,*),(q,4),(t,8)]-[q],[]-[(c,*),
(q,3),(s,6)]-[q]]
ATTACKS: [(q,a),(u,c)]
GRAPH: []
```

Example 5. Example of current visualisation using graphviz (from [1])



However these systems are not user-friendly enough to promote ABA as an easily accessible tool. Both systems require setting up and both are operated using Prolog which many users might not be comfortable with. This means that the systems are not readily accessible to users and the interface provided could be more useful. This project aims at creating a web-application that will enable users to easily and instantly interact with the systems in an online and more user-friendly environment.

2.8 ASPARTIX

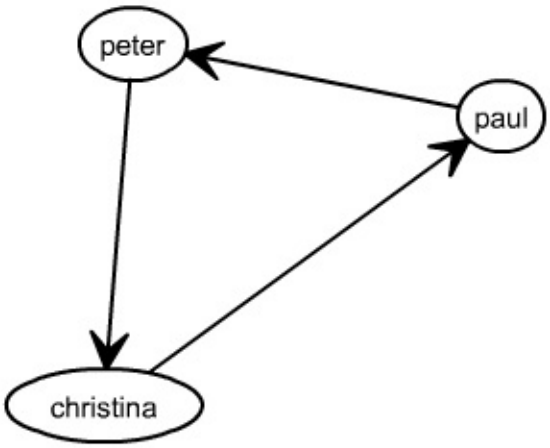
Perhaps the system that best resembles the desirable outcome of the project in terms of user experience and design is the ASPARTIX system (see [6]). ASPARTIX offers a web-application implementation that enables a user to enter a series of arguments and attack relationships between the arguments (see example 6), which is then processed and displayed diagrammatically to the user (see example 7). Our implementation should aim at providing a similar experience; allowing the user to implement their framework and displaying the results diagrammatically. However, ASPARTIX does not implement ABA instead it focuses more on Abstract Argumentation. Nonetheless, elements of the system such as its implementation as a web-application, the

user's ability to define a framework and the existence of a canvas on which the visualisation is displayed will have to be integrated in this project's web-application as well.

Example 6. Example of input in ASPARTIX (from [10])

```
arg(peter).
arg(paul).
arg(christina).
att(christina, paul).
att(peter, christina).
att(paul, peter).
```

Example 7. Example of visualisation of ASPARTIX (from [10])



2.9 Implementing Ideal semantics dispute derivations

Both Proxdd and Grapharg currently do not support analysing ABA according to ideal semantics. Nonetheless, the computational mechanisms do exist (see [5]). By extending the engines with this additional computational mechanism, we will allow our web-application to provide analysis under the ideal semantics with minimal additional development effort.

Essentially this form of dispute derivation is an adapted version of that of the admissible semantics. Formally it is defined as follows (according to [5]):

Definition 4. Let $\langle \mathcal{L}, \mathcal{R}, \mathcal{A}, \bar{} \rangle$ be an assumption based framework. Given a selection function, an *IB-dispute derivation of an ideal support* A for a

sentence α is a finite sequence of tuples

$\langle \mathcal{P}_0, \mathcal{O}_0, A_0, \mathcal{C}_0, \mathcal{F}_0 \rangle, \ldots, \langle \mathcal{P}_i, \mathcal{O}_i, A_i, \mathcal{C}_i, \mathcal{F}_i \rangle, \ldots, \langle \mathcal{P}_n, \mathcal{O}_n, A_n, \mathcal{C}_n, \mathcal{F}_n \rangle$
where

$\mathcal{P}_0 = \{\alpha\}$ $\quad \mathcal{A}_0 = \mathcal{A} \cap \mathcal{P}_0$ $\quad \mathcal{O}_0 = \mathcal{C}_0 = \mathcal{F}_0 = \{\}$
$\mathcal{P}_n = \mathcal{O}_n = \mathcal{F}_n = \{\}$ $\quad A = A_n$

and for every $0 \leq i < n$, only one σ in $\mathcal{P}_i$ or one S in $\mathcal{O}_i$ or one S in $\mathcal{F}_i$ is selected, and:

1. If $\sigma \in \mathcal{P}_i$ is selected then

   (a) if σ is an assumption then

   $\mathcal{P}_{i+1} = \mathcal{P}_i - \{\sigma\}$ $\quad A_{i+1} = A_i$ $\quad \mathcal{C}_{i+1} = \mathcal{C}_i$
   $\mathcal{O}_{i+1} = \mathcal{O}_i \cup \{\{\bar{\sigma}\}\}$ $\quad \mathcal{F}_{i+1} = \mathcal{F}_i$

   (b) if σ is not an assumption, then there exists some inference rule $\sigma \leftarrow R \in \mathcal{R}$ such that $\mathcal{C}_i \cap R = \{\}$ and

   $\mathcal{P}_{i+1} = \mathcal{P}_i - \{\sigma\} \cup (R - A_i)$ $\quad A_{i+1} = A_i \cup (A \cap R)$ $\quad \mathcal{C}_{i+1} = \mathcal{C}_i$
   $\mathcal{O}_{i+1} = \mathcal{O}_i$ $\quad \mathcal{F}_{i+1} = \mathcal{F}_i$

2. If S is selected in $\mathcal{O}_i$ then σ is selected in S_u and

   (a) if σ is an assumption, then

      i. either σ is ignored, i.e.

      $\mathcal{O}_{i+1} = \mathcal{O}_i - \{S\} \cup \{m(\sigma, S)\}$ $\quad \mathcal{P}_{i+1} = \mathcal{P}_i$ $\quad A_{i+1} = A_i$
      $\mathcal{C}_{i+1} = \mathcal{C}_i$ $\quad \mathcal{F}_{i+1} = \mathcal{F}_i$

      ii. or $\sigma \notin A_i$ and $\sigma \in \mathcal{C}_i$ and

      $\mathcal{O}_{i+1} = \mathcal{O}_i - \{S\}$ $\quad \mathcal{P}_{i+1} = \mathcal{P}_i$ $\quad A_{i+1} = A_i$
      $\mathcal{C}_{i+1} = \mathcal{C}_i$ $\quad \mathcal{F}_{i+1} = \mathcal{F}_i \cup \{u(S)\}$

      iii. or $\sigma \notin A_i$ and $\sigma \notin \mathcal{C}_i$ and

         A. if $\bar{\sigma}$ is not an assumption, then

$\mathcal{O}_{i+1} = \mathcal{O}_i - \{S\}$ $\mathcal{P}_{i+1} = \mathcal{P}_i \cup \{\bar{\sigma}\}$ $A_{i+1} = A_i$
$\mathcal{C}_{i+1} = \mathcal{C}_i \cup \{\sigma\}$ $\mathcal{F}_{i+1} = \mathcal{F}_i \cup \{u(S)\}$

B. if $\bar{\sigma}$ is an assumption, then

$\mathcal{O}_{i+1} = \mathcal{O}_i - \{S\}$ $\mathcal{P}_{i+1} = \mathcal{P}_i$ $A_{i+1} = A_i \cup \{\bar{\sigma}\}$
$\mathcal{C}_{i+1} = \mathcal{C}_i \cup \{\sigma\}$ $\mathcal{F}_{i+1} = \mathcal{F}_i \cup \{u(S)\}$

(b) if σ is not an assumption, then

$\mathcal{P}_{i+1} = \mathcal{P}_i$ $A_{i+1} = A_i$ $\mathcal{C}_{i+1} = \mathcal{C}_i$

$\mathcal{F}_{i+1} = \mathcal{F}_i \cup \{S - \{\sigma\} \cup R | \sigma \leftarrow R \in \mathcal{R} \text{ and } R \cap \mathcal{C}_i \neq \{\}\}$
$\mathcal{O}_{i+1} = \mathcal{O}_i - \{S\} \cup \{S - \{\sigma\} \cup R | \sigma \leftarrow R \in \mathcal{R} \text{ and } R \cap \mathcal{C}_i = \{\}\}$

3. If S is selected in $\mathcal{F}_i$ then *Fail(S)* and

$\mathcal{O}_{i+1} = \mathcal{O}_i$ $\mathcal{P}_{i+1} = \mathcal{P}_i$ $A_{i+1} = A_i$
$\mathcal{C}_{i+1} = \mathcal{C}_i$ $\mathcal{F}_{i+1} = \mathcal{F}_i - \{S\}$

Fail(S) is computed as follows:

Definition 5. Let $\langle \mathcal{L}, \mathcal{R}, \mathcal{A}, \bar{} \rangle$ be an assumption based framework. Given a selection function, a *Fail-dispute derivation* of a multiset of sentences S is a sequence $\mathcal{D}_0, \ldots, \mathcal{D}_n$ such that each $\mathcal{D}_i$ is a set of quadruples of the form $\langle \mathcal{P}, \mathcal{O}, A, \mathcal{C} \rangle$ where

$\mathcal{D}_0 = \{\langle S, \{\}, \mathcal{A} \cup S, \{\} \rangle\}$ $\mathcal{D}_n = \{\}$

and, for every $0 \leq i < n$, quadruple $Q = \langle \mathcal{P}, \mathcal{O}, A, \mathcal{C} \rangle$ is selected in $\mathcal{D}_i$ then either $\mathcal{P} \neq \{\}$ or $\mathcal{O} \neq \{\}$, and

1. If an element O from $\mathcal{O}$ is selected, then
   (a) if $O = \{\}$ then $\mathcal{D}_{i+1} = \mathcal{D}_i - \{Q\}$;
   (b) if $O \neq \{\}$ then let $\sigma \in O$ be the selected sentence in O:
      i. if σ is not an assumption then $\mathcal{D}_{i+1} = \mathcal{D}_i - \{Q\} \cup \{Q'\}$ where Q' is obtained from Q as in step (2.ii) of AB-dispute derivation;
      ii. if σ is an assumption then there are two cases:

Case 1: $\sigma \notin A$. Then $\mathcal{D}_{i+1} = \mathcal{D}_i - \{Q\} \cup \{Q_0, Q_1\}$ where Q_0 is obtained from Q as in step (2.i.a) and Q_1 is obtained from Q as in steps (2.i.b) or (2.i.c) (as applicable) of AB-dispute derivation;

Case 2: $\sigma \in A$. Then $\mathcal{D}_{i+1} = \mathcal{D}_i - \{Q\} \cup \{Q_0\}$ where Q_0 is obtained from Q as in step (2.i.a) of AB-dispute derivation;

2. If a $\sigma \in \mathcal{P}$ is selected, then

   (a) if σ is an assumption then $\mathcal{D}_{i+1} = \mathcal{D}_i - \{Q\} \cup \{Q'\}$ where Q' is obtained from Q as in step (1.i) of AB-dispute derivation;

   (b) if σ is not an assumption then $\mathcal{D}_{i+1} = \mathcal{D}_i - \{Q\} \cup \{Q'\}$ where there is a rule $\sigma \leftarrow R$ such that Q' is obtained from Q as in step (1.ii) of AB-dispute derivation.

The procedure defined above is explicitly explained in [5] and is used as the basis of the ideal semantics implementation. The web-application will now be able to analyse according to more semantics.

2.10 Decision Making with ABA

One of the most promising uses of argumentation is the potential of assisting decision making. This relies on being able to implement argumentation in a context that will allow it to compute the suitability of decisions. What makes argumentation exceptionally useful in the decision making context is that it not only proposes a decision, but it also provides the argumentation-based justification of it. This can be useful in real-life scenarios as it allows the user to understand the reasoning behind the decision taken. Research (see [11]) has proposed the use of Decision frameworks and Decision Functions that are mapped to ABA in order to compute the required decision.

Definition 6. A decision framework *is a tuple* $\langle D, A, G, T_{DA}, T_{GA} \rangle$, *consisting of:*

- a set of decisions $D = \{d_1, \ldots, d_n\}, n > 0$,
- a set of attributes $A = \{\alpha_1, \ldots, \alpha_m\}, m > 0$,
- a set of goals $G = \{g_1, \ldots, g_l\}, l > 0$, and
- two tables: T_{DA}, of size $(n \times m)$, and T_{GA}, of size $(l \times m)$, such that
  - for every $T_{DA}[i, j]^2, 1 \leq n, 1 \leq j \leq m, T_{DA}[i, j]$ is either 1, representing that a decision d_i has attributes a_j, or 0, otherwise;

– for every $T_{GA}[i,j], 1 \leq i \leq l, 1 \leq j \leq m, T_{GA}[i,j]$ is either 1, representing that goal g_i is satisfied by attribute a_j, or 0, otherwise.

In order to use ABA for decision the decision frameworks and decision functions need to be adapted to ABA. Computational Procedures have been identified for computing strongly dominant decisions, dominant decisions and weakly dominant decisions.

Definition 7. Given a decision framework $df = \langle D, A, G, T_{DA}, T_{GA} \rangle$, in which $|D| = n, |A| = m$ and $|G| = l$, the *strongly dominant ABA framework* corresponding to $\langle D, A, G, T_{DA}, T_{GA} \rangle$ is $df_s = \langle \mathcal{L}, \mathcal{R}, \mathcal{A}, \mathcal{C} \rangle$, where

- $\mathcal{R}$ is such that : for all $k = 1, \ldots, n; j = 1, \ldots, m$ and $i = 1, \ldots, l$:
  - if $T_{DA}[k,i] = 1$ then $d_k a_i \leftarrow$;
  - if $T_{GA}[j,i] = 1$ then $g_j a_i \leftarrow$;
  - $d_k g_j \leftarrow d_k a_i, g_j a_i$;
- $\mathcal{A}$ is such that: d_k, for $k = 1, \ldots, n; Nd_k g_j$, for $k = 1, \ldots, n$ and $j = 1, \ldots, m$;
- $\mathcal{C}$ is such that: $\mathcal{C}(d_k) = \{Nd_k g_1, \ldots, Nd_k g_n\}$, for $k = 1, \ldots, n$;

$\mathcal{C}(Nd_k g_j) = \{d_k g_j\}$, for $k = 1, \ldots, n$ and $j = 1, \ldots, m$.

Definition 8. Given $df = \langle D, A, G, T_{DA}, T_{GA} \rangle, |D| = n$, and $|A| = m$, let the corresponding strongly dominant ABA framework be $df_s = \langle \mathcal{L}, \mathcal{R}, \mathcal{A}, \mathcal{C} \rangle$, then the *dominant ABA framework corresponding to* df is $df_D = \langle \mathcal{L}, \mathcal{R}_\mathcal{D}, \mathcal{A}_\mathcal{D}, \mathcal{C}_\mathcal{D} \rangle$, where:

- $\mathcal{R}_\mathcal{D} = \mathcal{R} \cup Ng_j^{\bar{k}} \leftarrow Nd_1 g_j, \ldots, Nd_{k-1} g_j, Nd_{k+1} g_j, \ldots, Nd_N g_j\}$ for $k = 1, \ldots, n$ and $j = 1, \ldots, m$;
- $\mathcal{A}_\mathcal{D} = \mathcal{A}$;
- $\mathcal{C}_\mathcal{D}$ is $\mathcal{C}$ with $\mathcal{C}(Nd_k g_j) = \{d_k g_j\}$ replace by $\mathcal{C}(Nd_k g_j) = \{d_k g_j, Ng_j^{\bar{k}}\}$, for $k = 1, \ldots, n$ and $j = 1, \ldots, m$.

Definition 9. Given $df = \langle D, A, G, T_{DA}, T_{GA} \rangle, |D| = n$ and $|A| = m$, the *weakly dominant ABA framework corresponding to* df is $df_W = \langle \mathcal{L}, \mathcal{R}, \mathcal{A}, \mathcal{C} \rangle$, where

- $\mathcal{R}$ is such that: for all $k = 1, \ldots, n; j = 1, \ldots, m$ and $i = 1, \ldots, l$:
  - if $T_{DA}[k,i] = 1$ then $d_k a_i \leftarrow$;
  - if $T_{GA}[j,i] = 1$ then $g_j a_i \leftarrow$;
  - $d_k g_j \leftarrow d_k a_i, g_j a_i$;
    for all $r, k = 1, \ldots, n, r \neq k$; and $j = 1, \ldots, m$:

  - $Sd_rd_k \leftarrow d_rg_j, Nd_kg_j, NSd_kd_r$;
  - $\bar{S}d_kd_r \leftarrow d_kg_j, Nd_rg_j$;
- $\mathcal{A}$ is such that: d_k, for $k = 1, \ldots, n$;
  NSd_kd_r, for $r, k = 1, \ldots, n, r \neq k$;
  Nd_kg_j, for $k = 1, \ldots, n$ and $j = 1, \ldots, m$;
- $\mathcal{C}$ is such that: $\mathcal{C}(d_k) = \{Sd_1d_k, \ldots, Sd_{k-1}d_k, Sd_{k+1}d_k, \ldots, Sd_nd_k\}$, for $k = 1, \ldots, n$;
  $\mathcal{C}(NSd_kd_r) = \{\bar{S}d_kd_r\}$, for $r, k = 1, \ldots, n, r \neq k$;
  $\mathcal{C}(Nd_kg_j) = \{d_kg_j\}$, for $k = 1, \ldots, n$ and $j = 1, \ldots, m$.

By implementing the above mapping and the computational mechanisms provided we can enable our web-application to come closer to being decision-making tool.

2.11 Mapping AA to ABA

AA can be mapped to ABA and thus AA frameworks can be computed using ABA. Specifically the mapping is as follows (according to [7]):

Definition 10. Each AA framework $\mathcal{F} = (Arg, attacks)$ can be mapped onto a *corresponding ABA framework* $ABA(F) = \langle \mathcal{L}, \mathcal{R}, \mathcal{A}, \bar{} \rangle$ with

- $\mathcal{A} = Arg$;
- for any $\alpha \in \mathcal{A}, \bar{\alpha} = c(\alpha)$, with
  - $c(a) \notin \mathcal{A}$ and,
  - for $\alpha, \beta \in \mathcal{A}$, if $\alpha \neq \beta$ then $c(\alpha) \neq c(\beta)$;
- $\mathcal{R} = \{c(\alpha) \leftarrow \beta | (\beta, \alpha) \in attacks\}$;
- $\mathcal{L} = \mathcal{A} \cup \{c(\alpha) | \alpha \in \mathcal{A}\}$.

By implementing the mapping within the system, we can now allow it to take as an input an AA framework and evaluate it using the underlying ABA engines. This will make the web-application more flexible as it will be able to compute in accordance to different forms of argumentation.

3 Project Plan

"Failing to plan is planning to fail" — Winston Churchill

As with any project, planning and time management are imperative. To avoid getting overwhelmed by the workload required by the project, it has been preliminarily planned in the following way. Due to the agile nature of software development and the timeframe in which the implementation must be finalised, the following planning is preliminary and subject to change as the project moves on. For clarity purposes planning was split in three sections; Planning Implementation, Project Timeline, Planning Future extensions.

3.1 Planning Implementation

The application designed and implemented will take the form of a web-application. This will allow the system to be readily accessible to anyone over the internet, rather than having to be designed for a specific operating system and having to have the users to download and set-up the system. Such an approach will enhance the user-experience and make the system more maintainable as updates and additional features will have to be implemented just on the developers' side.

I have decided to use C# and the ASP.NET framework to develop the application. By using a widely supported framework I will ensure that the application functions correctly on a wide range of internet browsers. Additionally, the underlying Model-View-Controller software pattern of ASP.NET suits the problem at hand, as it will allow me to abstract the analysis aspect of the application from the visualisation aspect. This can have exceptional benefits in the future as different Views can be generated to represent other visualisation options in the future. For the visualisations themselves, I will be using the D3.js package that allows for easy interactive visualisations to be created. Lastly, the computational engines behind the application will be the existing Proxdd and Grapharg systems (described in section 2.6 and section 2.7) which will be invoked through SICSTUS Prolog.

3.2 Project Timeline

Below is the preliminary timetable for the tasks that need to be completed for the project. This timetable is open to revision in case the projects runs into trouble. Nonetheless, an underlying general timeframe exists and is defined by the milestones (shaded in grey). Ideally, the basic functionality required

for the project should be completed by the end of March, allowing April and May for implementing potential extensions (see section 2.9, section 2.10, section 2.11) and June for the final project write-up. These milestones can also act as roll-back positions. If the development of the extensions fail the system can always be rolled-back to its previous stable state (last achieved milestone).

Tasks	Start Date	End Date
Read research papers and explore existing applications	16/12/2013	20/1/2014
Decide on implementation languages and tools	16/12/2013	6/1/2014
Write Interim Report	16/1/2014	31/1/2014
Set-up tools and environments	1/2/2014	4/2/2014
Enable communication with engines over the internet	6/2/2014	7/2/2014
Implement Visualisation Canvas	8/2/2014	20/2/2014
Implement Input UI	8/2/2014	20/2/2014
Evaluate system with test examples	20/2/2014	27/2/2014
Finalise web-application	1/3/2014	10/3/2014
Basic Report write-up	1/3/2014	1/4/2014
Look into implementing ideal-semantics dispute derivation	1/4/2014	30/4/2014
Extended Report write-up	1/4/2014	30/4/2014
Look into implementing decision making with ABA	1/5/2014	1/6/2014
Look into implementing AA mapping to ABA	1/5/2014	1/6/2014
Final Report write-up	1/6/2014	17/6/2014

3.3 Planning Future Extensions

Having established the applications the project can, in the future, be extended to enhance the experience or add new functionality. For example, we could allow for the direct input of Decision Problems (see section 2.10) or we could extend the application to support AA argumentation as well by implementing the necessary mapping (see section 2.11). An attempt to these extensions has been scheduled and might be carried out if time allows it.

However, in addition to these planned extensions there are some interesting further enhancements that can be considered in the future. A creation of a stable and useful API for the underlying system would be of great use as it would allow for developers to make use of the system in there developing aspirations. This could provide the necessary platform for developers to launch Argumentation as a commercially viable tool. Additionally, this could lead to a further extension of the project that will involve the creation of real-life problem solvers that will use the argumentation engines. Due to the planned architecture of the system, this should be simple to do, especially with the inclusion of an API. Lastly, if decision tools are indeed designed using this system, then in the future it might be useful to consider make the web-application also available for hand-held devices. This will allow the tools to be used on the fly by users, making them even more accessible.

4 Evaluating Project

When designing and implementing a web-application there are several measures one must consider to evaluate the success of the project. The web-application is meant to be used by a wider range of users. Therefore, performance is not the only aspect of the application I must evaluate. User experience is equally important.

In order to evaluate the user experience and accessibility of the application the web-application should become available to pilot users early on. Through the feedback received, issues can be identified and the experience can be improved accordingly. Use of the web-application requires certain knowledge of argumentation. By creating a simple informative section on the website of the web-application the concept should be introduced to a level that would allow users to provide feedback on their interaction with the application.

Performance wise the system will be tested with a series of example frameworks, to ensure its performance both in validity and in computational speed. The existing systems do compute with a satisfactory speed. The web-application should not hinder this performance significantly. To ensure that the application performs well, the test cases will be vary in terms of complexity and size.

Lastly, certain automated tests will be used during the development phase to ensure that the system is robust against certain likely software issues. One of the aims of the project is for the application to be extendible in the future. The existence of these automated tests has the additional benefit of making the application more extendible by other developers in the future. By providing these tests, we allow the developers to ensure that the initial functionality of the application is not obstructed by any additional functionality they might introduce. Such, tools are important in ensuring that the application is extendible beyond the scope of this project.

References

[1] Dr. Robert Craven. Grapharg. *http://www.doc.ic.ac.uk/∼rac101/proarg/grapharg.html*.

[2] Dr. Robert Craven. Proxdd. *http://www.doc.ic.ac.uk/∼rac101/proarg/proxdd.html*.

[3] Robert Craven, Francesca Toni, and Matthew Williams. Graph-based disput derivations in assumption-based argumentation. In *TAFA 2013*.

[4] Phan Min Dung, Robert Kowalski, and Francesca Toni. Assumption-based argumentation. In *Argumentation in AI*.

[5] P.M. Dung, P. Mancarella, and F. Toni. Computing ideal sceptical argumentation. *Artificial Intelligence*, 171(1015):657 – 660, 2007. ¡ce:title¿Argumentation in Artificial Intelligence¡/ce:title¿.

[6] Uwe Egly, Sarah Alice Gaggl, and Stefan Woltran. Aspartix: Implementing argumentation frameworks using answer-set programming. In Maria Garcia de la Banda and Enrico Pontelli, editors, *Logic Programming, 24th International Conference, ICLP 2008, Udine, Italy, December 9-13 2008, Proceedings*, volume 5366 of *Lecture Notes in Computer Science*, pages 734–738. Springer, 2008.

[7] Francesca Toni. Reasoning on the web with assumption-based argumentation. In *8th Reasoning Web Summer School volume 7487 of Lecture Notes in Computer Science*.

[8] Francesca Toni. A generalised framework for dispute derivations in assumption-based argumentation. *Artificial Intelligence*, 195(0):1 – 43, 2013.

[9] Francesca Toni. A tutorial on assumption-based argumentation. *Argument & Computation*, 2013.

[10] TU Wien. Aspartix. *http://www.dbai.tuwien.ac.at/proj/argumentation/systempage/*.

[11] Fan Xiuyi and Toni Francesca. Decision making with assumption-based argumentation. In *TAFA 2013*.