Argumentation Theory

Nonmonotonic Reasoning

BACHELOR'S THESIS

submitted in partial fulfillment of the requirements for the degree of

Bachelor of Science

in

Software- and Information Engineering

by

Christoph Groß, 1025119

to the Faculty of Informatics
at the Vienna University of Technology

Advisor: Ao.Prof.Dipl.-Ing.Dr.techn. Christian Georg Fermüller

Contents

List of Figures

Statutory Declaration

Hereby I assure that this thesis is a result of my personal work and that no other than the indicated aids have been used for its completion.
Furthermore I assure that all the quotations and statements, that have been inferred literally or in a general manner from published or unpublished writings, are marked as such.
Beyond this I assure that the work has not been used, neither completely nor in parts, to pass any previous examination.

Vienna, June 2, 2015

Christoph Groß

Acknowledgements

$D := \{Friends, Family, University, christoph\}$
$Friends = \{\text{Max,Pia,Lea,Tobi,Johannes}\}$,
$Family = \{\text{Sabine,Gerald,Angelika,Johann,Johanna}\}$,
$University = \{\text{Prof.Fermüller}\}$

$$\forall x \in D : (proofread(x) \vee listenToWhining(x) \vee help(x)) \implies thanks(christoph, x)$$

Abstract

Throughout this thesis, I am going to explore the applications of argumentation theory in general and particularily in the context of nonmonotonic reasoning.
First off, I will introduce the necessary formal definitions, give a few easy examples as clarification and draw the theoretical conclusions.
Then I try to show the link between argumentation frameworks and default theory as an example of how argumentation theory can be used for dealing with nonmonotonic reasoning.
In the next step, the reader receives information about things that have not been dealt with in this thesis, some examples of real-life applications and also pointers to the needed sources to get a deeper understanding of the subject.
Finally I will append the documentation of an application specifically written during the course of this thesis, which purpose is to graphically help understanding the subject.

This thesis, including all the cited sources and the application (source code as well as an executable jar file) can be found on github. Should any questions or ideas for improvement arise, do not hesitate to write to e1025119@student.tuwien.ac.at

Chapter 1

Introduction

1.1 Argumentation Theory

In a human context, we (hopefully) argue to reach a sound conclusion. We do that by claiming statements based on premisses, and usually, due to different perception of subjects as well as other factors, these statements tend to vary in their degree of believability.

Argumentation theory in general, is the interdisciplinary study of how conclusions can be reached through logical reasoning. Since this is a very wide field, I can not hope to discuss all of it here. The wikipedia article provides a good overview for interested readers.

This thesis focuses on the logical approach made by Dung [1] in his milestone paper from 1995, the so called "argumentation frameworks". Argumentation Frameworks try to formalize arguments and their semantics. To this purpose, there are two main components. First, there are arguments modeled as formulas and second, there are attack relations between said arguments.

1.1.1 Argumentation Frameworks

Definition 1.1 (Argumentation Framework - [1])
An argumentation framework $\boldsymbol{AF}$ *is defined as a tuple, consisting of a non-empty set of arguments (*$\boldsymbol{AR}$*) and a possibly empty set of attack relations (*$\boldsymbol{att}$*).*
Attack relations are tuples of two arguments where the first argument attacks the second one.
$\boldsymbol{AF := \langle AR, att \rangle}$ ■

The following example shows that an argumentation framework can either be written in a form according to this definition, or visualized as a directed graph.

Example 1.1
$\boldsymbol{s}$ =”Superman is the best superhero!”, $\boldsymbol{b}$ =”Batman is way cooler..”,
$\boldsymbol{a}$ =”Only Aquaman rules over the seas.”, $\boldsymbol{r}$ =”Robin is Batman’s sidekick.”
$\boldsymbol{AR} = \{\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{s}\}$, $\boldsymbol{att} = \{(\boldsymbol{b}, \boldsymbol{s}), (\boldsymbol{b}, \boldsymbol{a}), (\boldsymbol{a}, \boldsymbol{s}), (\boldsymbol{a}, \boldsymbol{b}), (\boldsymbol{s}, \boldsymbol{b}), (\boldsymbol{s}, \boldsymbol{a}), (\boldsymbol{b}, \boldsymbol{r})\}$ ■

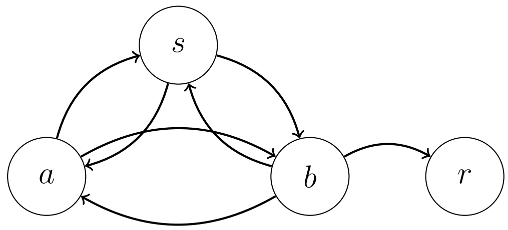

Figure 1.1: graph of AF (example 1.1)

1.1.2 Properties of (Sets of) Arguments

The following definitions are originally found in [1].

Definition 1.2 (Conflict-freeness)
A set $\boldsymbol{S}$ of arguments is said to be conflict-free, if for any argument $\boldsymbol{a} \in \boldsymbol{S}$, there is no other argument $\boldsymbol{b} \in \boldsymbol{S}$ so that $(\boldsymbol{a}, \boldsymbol{b}) \in \boldsymbol{att}$. ■

Definition 1.3 (Acceptability)
An argument $\boldsymbol{a} \in \boldsymbol{AR}$ is acceptable with respect to a set $\boldsymbol{S}$, iff $\boldsymbol{S}$ attacks any argument $\boldsymbol{b} \in \boldsymbol{AR}$ that attacks $\boldsymbol{a}$. ■

Remark
If for said argument $\boldsymbol{a}$ is not being attacked by any $\boldsymbol{b} \in \boldsymbol{AR}$, then it is acceptable with even the empty set. ■

Definition 1.4 (Admissibility)
A conflict-free Set $\boldsymbol{S}$ is said to be admissible, iff each argument $\boldsymbol{a} \in \boldsymbol{S}$ is acceptable with respect to $\boldsymbol{S}$. Often, $\boldsymbol{S}$ is also called an admissible extension. ■

Remark
In example 1.1, the admissible extensions would be $\boldsymbol{E_A} = \{\emptyset, \{a\}, \{b\}, \{s\}, \{r, a\}, \{r, s\}\}$. ■

These are the basic properties that can be fulfilled by arguments or sets of arguments. Based on them, we will now introduce a few more interesting definitions. For instance, argumentation frameworks know a number of different extensions, which can be used to capture the semantics of their subsets.

If not explicitly marked, all the following extensions can also be originally found in [1].

Remark (**Notation of extensions**)
From here on, we will use $\boldsymbol{E_i}$ *to denote an arbitrary but fixed extension of type* $\boldsymbol{i}$*, while* $\boldsymbol{E_I}$ *denotes the set of all extensions of type* $\boldsymbol{i}$*.*
Also, if we want to talk about the set of all extensions of one specific type, we will use $\boldsymbol{i(AF)}$*.* ■

The first of those extensions takes a rather gullible approach to things; the so called *preferred extension* is defined as follows:

Definition 1.5 (Preferred Extensions)
$\boldsymbol{E_p}$ *is a preferred extension of* $\boldsymbol{AF}$*, iff* $\boldsymbol{E_p}$ *is a maximum admissible set of* $\boldsymbol{AF}$*.* ■

Remark
In example 1.1, the preferred extensions would be $\boldsymbol{E_P = \{\{r, a\}, \{r, s\}\}}$*.* ■

As you can see, the preferred extension is made up of the biggest, conflict-free set of arguments, that can be defended against all other arguments not in the set.
The following lemma will be useful later on.

Lemma 1.1 ([1])
For every AF, there exists at least one preferred extension. ■

Proof
The proof will be depending on two different properties of admissible sets:

1. *The empty set is always admissible.*
2. *Given an admissible Set* $\boldsymbol{S}$ *and an argument* $\boldsymbol{a}$*, which is acceptable with respect to* $\boldsymbol{S}$*, the unification* $\boldsymbol{S \cup a}$ *is also admissible.*

(1) *directly follows from the definition 1.4*
(2) *We just have to show, that* $\boldsymbol{S \cup a}$ *is conflict-free. To achieve that, we assume the contrary: There exists an argument* $\boldsymbol{b \in S}$ *so that either* $\boldsymbol{(a, b)}$ *or* $\boldsymbol{(b, a)}$ *holds. There has to be an argument* $\boldsymbol{c \in S}$*, so that either* $\boldsymbol{(c, a)}$ *or* $\boldsymbol{(c, b)}$*. We know, that* $\boldsymbol{S}$ *is conflict-free, so* $\boldsymbol{(c, b) \implies \lightning}$
On the other hand, $\boldsymbol{c}$ *and* $\boldsymbol{a}$ *are both acceptable with respect to* $\boldsymbol{S}$*;* $\boldsymbol{(c, a) \implies \lightning}$*.* ■

Another approach would be the *stable extension*, which is defined as follows:

Definition 1.6 (Stable Extensions)
The conflict-free set $\boldsymbol{E_s}$ *is a stable extension of* $\boldsymbol{AF}$*, iff* $\boldsymbol{E_s}$ *attacks each argument* $\boldsymbol{a \notin E_s}$*; which is equivalent to* $\boldsymbol{\{a | a}$ *is not attacked by* $\boldsymbol{E_s\}}$*.* ■

Remark
In example 1.1, the stable extensions would be $\boldsymbol{E_S = \{\{b\}, \{r, a\}, \{r, s\}\}}$*.* ■

Definition 1.7 (Complete Extensions)
The admissible set $\boldsymbol{E_c}$ *is a complete extension of* $\boldsymbol{AF}$*, iff every argument which is acceptable with respect to* $\boldsymbol{E_c}$*, is included in* $\boldsymbol{E_c}$*.* ■

Remark
In example 1.1, the complete extensions would be $\boldsymbol{E_C = \{\emptyset, \{b\}, \{r, a\}, \{r, s\}\}}$*.* ■

It is easy to see that many different complete extensions can exist, depending on E_c. That leads directly to the *grounded extension*.

Definition 1.8 (Grounded Extensions - [2])
The intersection of all complete extensions is called the grounded extension.
$\boldsymbol{E_g := \bigcap comp(AF)}$ ■

Remark
In example 1.1, the only grounded extension would be $\boldsymbol{E_g} = \emptyset$*.* ■

Definition 1.9 (Semi-stable Extensions - [3])
A set $\boldsymbol{E_{SS}}$ *is a semi-stable extensions, if it is a complete extension and the set* $\boldsymbol{E_{ss} \cup \{a \in AR | E_{SS}}$ *attacks* $\boldsymbol{a\}}$ *is maximal (with respect to set inclusion).* ■

Example 1.2
A very good example, specifically showing that not every semi-stable extension is also stable, was provided in [3].
In the following graph, the set $\boldsymbol{E_{SS} = \{b, d\}}$ *is semi-stable but not stable.* ■

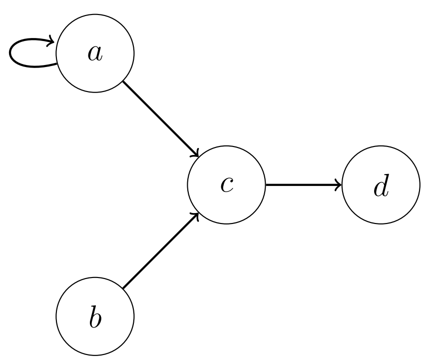

Figure 1.2: semi-stable but not stable (example 1.2)

Definition 1.10 (Ideal Extensions - [4])
A set $\boldsymbol{E_I}$ is an ideal extension, if it is the maximal admissible set, that is contained in every preferred extension. ■

Example 1.3
The following example illustrates ideal extensions (and how they differ from e.g. grounded extensions).
In the following graph, the set $\boldsymbol{E_I = \{b\}}$ is an ideal extension. This example can also be found in [4]. ■

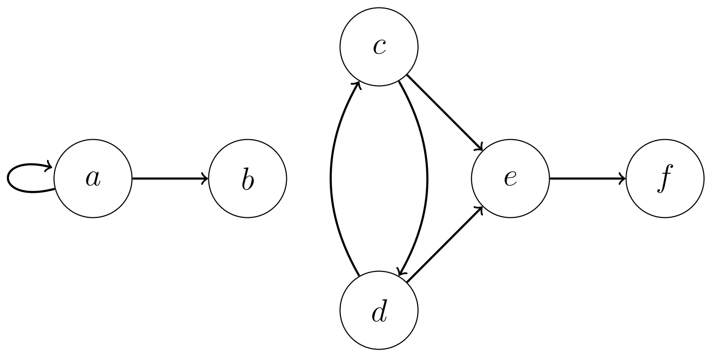

Figure 1.3: ideal extension (example 1.3)

As we can see, the ideal extension $\boldsymbol{E_I = \{b\}}$ is a subset of every preferred extension ($\boldsymbol{\{b, cf\}, \{b, d, f\}}$).

1.1.3 Examples of different Extensions

Contrary to the previous section, where different aspects of one example were used to accompany the definitions of extensions, here I am going to provide examples which simply are interesting in their respective results.

Example 1.4
$\boldsymbol{s_1}$ =*"Superman is the best superhero!"*, $\boldsymbol{b_1}$ =*"Batman is way cooler.."*,
$\boldsymbol{s_2}$ =*"But Superman is stronger."*, $\boldsymbol{b_2}$ =*"Batman possesses genius-level intellect."*
$\boldsymbol{AR = \{s_1, s_2, b_1, b_2\}}$, $\boldsymbol{att = \{(b_1, s_1), (s_2, b_1), (b_2, s_2)\}}$ ■

This argumentation framework could also be depicted by following graph.

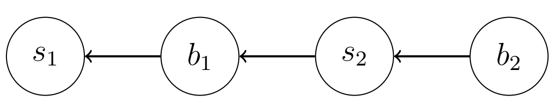

Figure 1.4: graph of AF (example 1.4)

The extensions of this example would be

- $\boldsymbol{E_P = E_S = E_C = E_G = \{\{b_1, b_2\}\}}$

The next example is going to be a little more interesting in its outcome.

Example 1.5
$\boldsymbol{s}$ =*"Superman was born a hero."*, $\boldsymbol{b}$ =*"Batman trained hard to become a hero."*
$\boldsymbol{AR = \{s, b\}}$, $\boldsymbol{att = \{(s, b), (b, s)\}}$ ■

This examples graph looks like this.

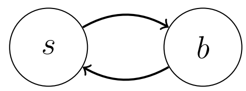

Figure 1.5: graph of AF (example 1.5)

Following extensions can be calculated

- $\boldsymbol{E_P = E_S = \{\{s\}, \{b\}\}}$
- $\boldsymbol{E_C = \{\emptyset, \{s\}, \{b\}\}}$
- $\boldsymbol{E_G = \{\emptyset\}}$

The following example (on the contrary to the above) has different preferred and stable extensions.

Example 1.6
$\boldsymbol{b}$ =*"Batman is Clark Kent."*
$\boldsymbol{AR = \{b\}}$, $\boldsymbol{att = \{(b, b)\}}$ ■

The corresponding graph looks like follows.

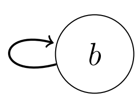

Figure 1.6: graph of AF (example 1.6)

For this example, the extensions would look as follows.

- $\boldsymbol{E_P = E_C = E_G = \{\emptyset\}}$
- $\boldsymbol{E_S = \emptyset}$

Example 1.7
Last but not least, we are going to take a look on a slightly more complex example, that gives a deeper insight in the relations between the extensions. It should be considered along with ***Figure 2.1****.*
In the graph below, following different extensions can be oserved:

- *semi-stable extension* $\boldsymbol{E_{ss} = \{d, e\}}$
- *preferred extension* $\boldsymbol{E_p = \{b, c\}}$
- *complete extension* $\boldsymbol{E_c = \{j\}}$
- *grounded extension* $\boldsymbol{E_g = \emptyset}$

■

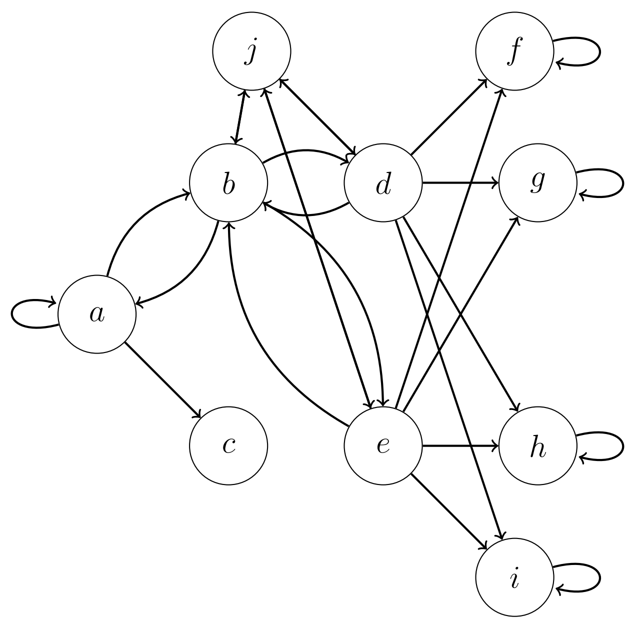

Figure 1.7: relations of extensions (example 1.7)

Remark
Now it comes to attention that there are no stable and semi-stable extensions in the above example at the same time. That is because it is not possible for distinctive stable and semi-stable extensions to coexist in a single AF. We will take a quick look at the reasons for this particular relation. ■

Proof
Assume the contrary, i.e. there exists a stable extension $\boldsymbol{E_s}$ *and a (only) semi-stable extension* $\boldsymbol{E_{ss}}$*.*
From the definitions of stable and semi-stable extensions follows
$\boldsymbol{E_s \cup \{a | E_s}$ *attacks* $\boldsymbol{a\} = AR}$*. Since every semi-stable extension has to fulfill*
$\boldsymbol{E_{ss} \cup \{a | E_{ss}}$ *attacks* $\boldsymbol{a\}}$ ***is maximal with respect to set inclusion****, it is clear that* $\boldsymbol{E_{ss}}$ *has to attack every argument outside of itself, in order to achieve maximum set inclusion.*
This directly contradicts the assumption $\boldsymbol{E_s \notin \{E|}$ *E is a stable extension* $\boldsymbol{\}} \implies \lightning$*.* ■

Chapter 2

Conclusions and Implications

Now that we have established the necessary definitions, it is time to travel further down the rabbit hole and explore the meaning of what we just defined.

2.1 Relations between different Extensions

The following, simple tree graph (borrowed from [5]) shows the hierarchy of extension types better than words could ever describe it.

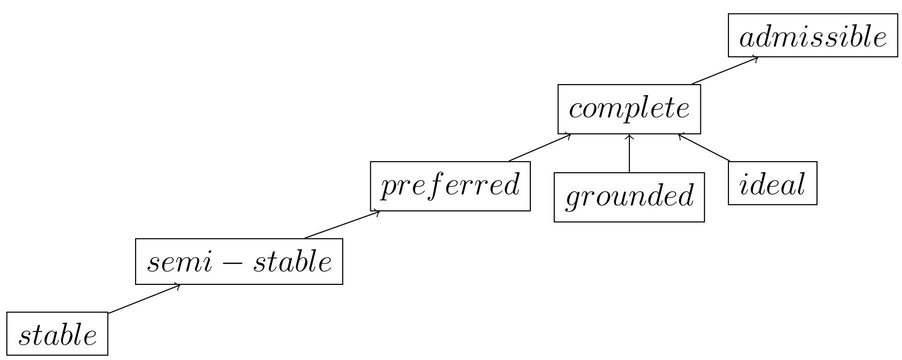

Figure 2.1: hierarchy of extension types - An arrow from X to Y denotes that an X-extension is also a Y-extension.

Now I want to show a few specific relations between those aforementioned extensions, since they will be reused in later proofs.

Lemma 2.1 ([1])

1. *Every preferred extension is a complete extension, but not vice versa.*
2. *The grounded extension is the least (with respect to set inclusion) complete extension*

■

Proof (1)
It is obvious that every preferred extension (maximum admissible set) is a complete extension. So we only need to show that the other way round is not true.

Assumption: Every complete extension is also a preferred extension.
Let $\boldsymbol{AR} = \{\boldsymbol{a}, \boldsymbol{b}\}$ *and* $\boldsymbol{att} = \{(\boldsymbol{a}, \boldsymbol{b}), (\boldsymbol{b}, \boldsymbol{a})\}$.
Then it is easy to see, that $\boldsymbol{E_P} = \{\{\boldsymbol{a}\}, \{\boldsymbol{b}\}\}$ *but* $\boldsymbol{E_C} = \{\boldsymbol{\emptyset}, \{\boldsymbol{a}\}, \{\boldsymbol{b}\}\}$
$\implies \exists \boldsymbol{E_c} : \boldsymbol{E_c} \in \boldsymbol{E_C} \wedge \boldsymbol{E_c} \notin \boldsymbol{E_P} \implies \lightning$ ■

Proof (2)
Proof for this lemma follows directly from definition 1.8. ■

2.2 Properties of Argumentation Frameworks

Up to now, we have only talked about sets of arguments and their properties. Now we will focus on the whole argumentation framework and make a connection between its properties and extensions.

2.2.1 Well-Founded Argumentation Frameworks

The first new property establishes a coherence between preferred, grounded and stable extensions.

Definition 2.1 (Well-Founded Argumentation Frameworks)
An argumentation framework is considered well-founded, iff there exists no infinit sequence of arguments $A_0, A_1, \ldots, A_n, \ldots$ *where each* A_{i+1} *attacks* A_i. ■

Remark
This infinite sequence of arguments can be formed by a finite set of arguments (for instance $AR = \{a, b\}$, $att = (a, b), (b, a)$*).* ■

Although it might not be the most intuitive observation, following lemma can be extracted from this definition.

Lemma 2.2 ([1])
Every well-founded AF has exactly one complete extension, which is also a preferred, grounded and stable extension. ■

Proof
Assumption: $\exists \boldsymbol{AF}$ **:** *AF is well-founded and* $\boldsymbol{E_p} = \boldsymbol{E_c} = \boldsymbol{E_g} \notin \boldsymbol{E_S}$*. We further assume, that* $\exists \boldsymbol{a} \in \boldsymbol{AR} \backslash \boldsymbol{E_g}$ *and* $\nexists \boldsymbol{b} \in \boldsymbol{E_g} : (\boldsymbol{b}, \boldsymbol{a}) \in \boldsymbol{att}$*.*

If $\boldsymbol{a}$ *is not acceptable with respect to* $\boldsymbol{E_g}$*, this means that there has to be another argument* $\boldsymbol{c} \in \boldsymbol{AR} \backslash \boldsymbol{E_g}$*, which is also not attacked by* $\boldsymbol{E_g}$*.*
Therefore an infinite sequence of arguments like $\boldsymbol{a}$ *and* $\boldsymbol{c}$ *has to exist* $\implies \lightning$. ■

2.2.2 Coherent Argumentation Frameworks

If an AF has a preferred extension, which is not a stable one too, that indicates some sort of "anomaly" in the framework (see example 1.6).
So what we want to do next, is to find out what criteria has to be met so that preferred and stable extension coincide.

Definition 2.2 (Coherent Argumentation Frameworks)
An AF is called coherent, if $\boldsymbol{E_P} = \boldsymbol{E_S}$*.* ■

Remark
It follows directly from the definition, that there always exists at least one stable extension in a coherent AF. ■

Chapter 3

Argumentation Theory and Nonmonotonic Reasoning

One reason why argumentation frameworks are important and useful is the fact that they can easily be used to model other logical constructs, often making working with those less complex.
For instance, the nonmonotonic "Default Logic", which is briefly described in the following section.
Since all the definitions in this next section were taken from [6], that paper provides more insight to the interested reader.

3.1 Default Logic

In classical logic, it is not possible to conclude new knowledge from uncertain/unknown information.
One scenario for instance, would be the modeling of a default property, which could be assigned to any object as long as we do not have any contradicting facts.
This is where default logic comes in handy.

Definition 3.1 (Default Logic)
A default theory is a pair $\langle \boldsymbol{W}, \boldsymbol{D} \rangle$. $\boldsymbol{W}$ *is the knowledge base, consisting of logical formulas,* $\boldsymbol{D}$ *is a set of default rules. Each default rule has following form:*

$$\frac{\boldsymbol{Prerequisite : Justification_1, \ldots, Justification_n}}{\boldsymbol{Conclusion}}$$

■

A simple (and maybe overused) example is the following.

Example 3.1

$$\boldsymbol{D} = \frac{\boldsymbol{Bird(tweety) : Flies(Tweety)}}{\boldsymbol{Flies(Tweety)}}$$

$\boldsymbol{W_1 = \{Bird(tweety)\}}$
$\boldsymbol{W_2 = \{Bird(tweety), Penguin(tweety), Penguin(X) \implies \neg Flies(X)\}}$ ■

As we can see, as long as no facts, that negate one of the justifications, can be deduced from our knowledge base, we can infere the conclusion that Tweety can fly (which would be the case with $\boldsymbol{W_1}$).
On the other hand (with $\boldsymbol{W_2}$), we could deduce $\boldsymbol{\neg Flies(Tweety)}$, therefore the Justifications would not be met and the default would be overruled by the more important fact.

Furthermore we have to define the so-called "Reiter's extension" (or R-extension).

Definition 3.2
A R-extension of a default theory $\boldsymbol{\langle W, D\rangle}$ is a set of formulas $\boldsymbol{E_R}$, that satisfies following conditions:

- $\boldsymbol{E_R = \bigcup\{W_i | i}$ *is a natural number*$\boldsymbol{\}}$
- $\boldsymbol{W_0 = W}$
- $\boldsymbol{W_{i+1} = Th(W_i) \cup \{w | \exists(p : j_1, \ldots, j_k/w)}$ *in* $\boldsymbol{D}$ *so that* $\boldsymbol{\{j_n\} \cup E}$ *consistent for* $\boldsymbol{k \geq n \geq 1,}$ *and* $\boldsymbol{p \in W_i\}}$

$Th(T)$ denotes the first order closure of a theory T. ■

3.2 Default Logic & Argumentation Frameworks

As it is pointed out in 3.1 of [1], a default theory $\boldsymbol{T = \langle W, D\rangle}$ can be interpreted as an argumentation framework $\boldsymbol{AF_d = \langle AR, Att\rangle}$.
From here on out, we let $\boldsymbol{just(D)}$ denote the set of justifications of a default theory.

The following two criteria are necessary to build the interpretation:

- $\boldsymbol{AF_d = \{(K, k) | K \subseteq just(D)\}}$, so that $\boldsymbol{K}$ is a support of the well-formed, closed formula $\boldsymbol{k}$ with respect to $\boldsymbol{T}$
- $\boldsymbol{(K, k)}$ attacks $\boldsymbol{(K', k')}$ iff $\boldsymbol{\neg k \in K'}$

The following lemma shows that this interpretation is not just a coincidence.

Lemma 3.1 ([1])
Let $\boldsymbol{E_a}$ be an admissible subset of $\boldsymbol{AF_d}$ and $\boldsymbol{H = \bigcup\{K | (K, k) \in E_a\}}$. Then $\boldsymbol{T, H}$ holds, iff $\boldsymbol{T}$ is consistent. ■

Proof
From left to right, this should be obvious, so we are only going to take a look at the second part of the proof. To achieve this, we assume the contrary:
There exists a subset $\boldsymbol{K}$ of $\boldsymbol{H}$, so that $\boldsymbol{T, K \vdash \bot}$. Thus, for every $\boldsymbol{k}$, $\boldsymbol{(K, k) \in AR}$. Also, let $\boldsymbol{(K', k') \in E_a}$ so that $\boldsymbol{K'}$ is not empty. Now $\boldsymbol{(K, k)}$ represents an attack on $\boldsymbol{(K', k')}$.
Given the admissibility of $\boldsymbol{E_a}$, there has to be some argument $\boldsymbol{a = (H', h') \in E_a}$, so that $\boldsymbol{\neg h' \in K}$; in other words, since $\boldsymbol{(K, k), (K', k') \in E_a}$ and the admissibility of
$\boldsymbol{E_a}$, the set cant be conflict-free $\implies \lightning$ ■

Now that we have established the possibility to interprete default theories as argumentation frameworks, we want to examine this relationship more closely. There is a correspondence between the R-extensions of a default theory and the stable extensions of an argumentation framework. To show this, we need some more definitions:

Definition 3.3 ([1])
Let $\boldsymbol{S}$ *be a first order theory and* $\boldsymbol{S'}$ *be a set of arguments of* $\boldsymbol{AF_d}$*. Then, we define:*

- $arg(S) = \{(K,k) \in AR | \forall j \in K : \{j\} \cup S \text{ } is\ consistent\}$
- $flat(S') = \{k | \exists (K,k) \in S'\}$

■

Together with the definition of the R-extension, the following lemmata become obvious:

Lemma 3.2 ([1])
If $\boldsymbol{T}$ *is a default theory and* $\boldsymbol{E}$ *is a first order theory. Then,* $\boldsymbol{E}$ *is a R-extension of* $\boldsymbol{T}$ *iff* $\boldsymbol{E = flat(arg(E))}$. ■

Lemma 3.3 ([1])
$\boldsymbol{T = \langle W, D \rangle}$ *be a default theory,* $\boldsymbol{E}$ *a R-extension of* $\boldsymbol{T}$ *and* $\boldsymbol{E'}$ *a stable extension of* $\boldsymbol{AF_d}$*. Then*

1. $\boldsymbol{arg(E)}$ *is a stable extension of* $\boldsymbol{AF_d}$
2. $\boldsymbol{flat(E')}$ *is a R-extension of* $\boldsymbol{T}$

■

Apparently, the preferred extension semantics of $\boldsymbol{AF_d}$ generalizes the R-extension semantics of $\boldsymbol{T}$. Also it captures the intuitive idea of default logic more naturally and thus makes it a bit easier to model and understand. This was just one of the many applications of argumentation frameworks, the interested reader could either get more examples in [1] or head on to the "Further Topics" chapter for more information.

Chapter 4

Further Topics

At this point, some interested reader might have noticed that we have been dealing with the abstract concept of argumentation frameworks almost exclusively. Since this is a bachelor's thesis, I had to focus on one single aspect in order to not get lost in the process of writing it.
Therefore, the idea of this fourth chapter is providing you with pointers to related subjects, sources of deeper understanding and real-life applications to tamper with.

4.1 Structure and Extraction of Arguments

Two very interesting topics are the structure of arguments and how they are extracted. During this paper, we only used arguments as abstract objects with no underlying structure. If we were trying to solve real world problems by implementing them as an argumentation framework, these two things would be necessary though. The general structure of arguments can be describes as it is done in the very good wikipedia article:

arguments typically consist of

- a set of assumptions, often called premises or support
- a conclusion, often called claim

To gain more insight on the structure of arguments, one could also read [7] or [8].
If the reader is more interested in how arguments are extracted from knowledge, a process which ideally follows certain rules, then [9] could be a good source.

4.2 Real-life Applications

Since we have already established, that argumentation frameworks can be used to model other logical constructs, it is clear that they can also be a useful tool in various proofs.
This is, of course, of very little use in the daily life of a software engineer. In general, the areas where logical applications shine, are for instance **planning**, **scheduling** or **software verification**, all areas with a high potential to save some company a lot of money.
If one would like to gain more knowledge on these topics, I would like to point him towards the respective wikipedia articles as a starting point.

An interesting tool for creating, exploring and evaluating argumentation frameworks is being developed and maintained from the *"database and artificial intelligence group"* at Vienna Technical University (ASPARTIX).
A much simpler (and less powerful) approach is provided within the next chapter.

Chapter 5

Documentation - KAFFEE

As mentioned in the abstract, a software tool was written to accompany the reader through this paper and provide the necessary environment to playfully learn more about the several extensions and their relationships.
The **K**AFFEE **A**rgumentation **F**ramework **F**or **E**ngineers and **E**nthusiasts offers following functions:

- Defining new argumentation frameworks
- Import prepared examples
- Select different types of extensions
- Visualize whole AFs as well as subsets
- Step-by-Step function that explains how an admissible set is created

The general idea of how to use **KAFFEE** is simple. The tool is made up of three tabs, which are sequentially dependant on the one before, as shown in the following graph.

Tab ”Argumentation Framework” → **Tab ”Extensions”** → **Tab ”Step By Step”**

Figure 5.1: Workflow in KAFFEE

That requires the user to work in tab #1 first, the ”Argumentation Framework” tab.
Here, the user has to create/import an argumentation framework.
Creating a framework can be achieved by submitting arguments and (optional) attack relations. To do so, the user first has to enter a reference name and text (in case of an argument) or two existing arguments (in case of an attack relation) and then click the corresponding submit button.
The faster way is to import one of the existing example frameworks. This can be easily done, the user only has to click **Examples** in the menu bar and choose one of the provided examples.

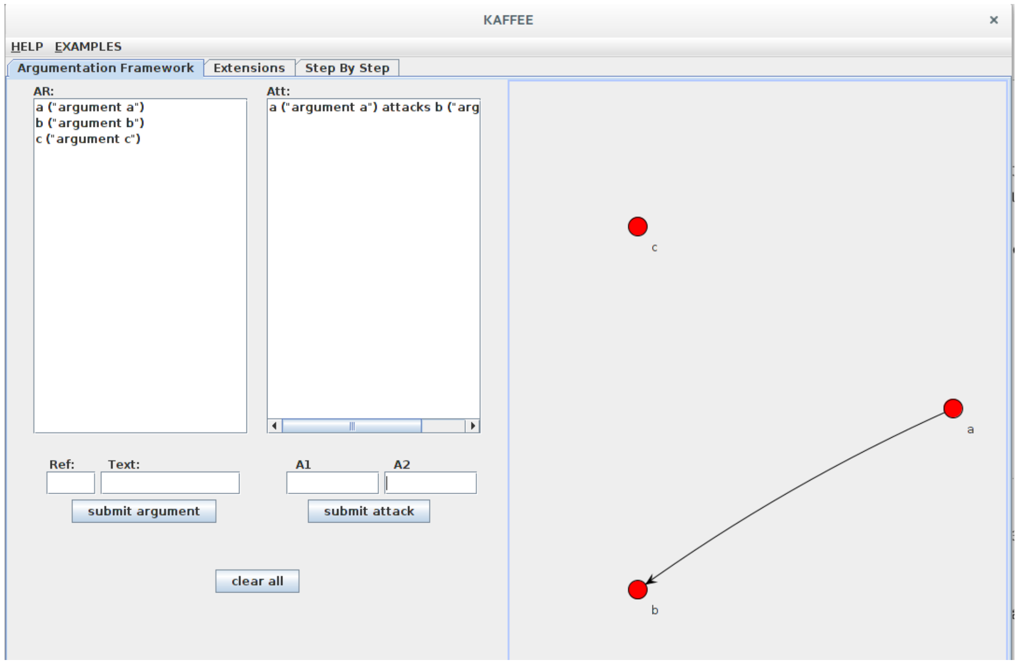

Figure 5.2: Tab "Argumentation Framework"

Once that is done, the user can either have a look at the visual representation of his/her framework, or continue and switch to the "Extensions" tab. In this tab, one can choose from a list of possible extensions to be calculated for the given AF. This time, the visual representation on the right will provide coloring of the selected subset, depending on the chosen extension.

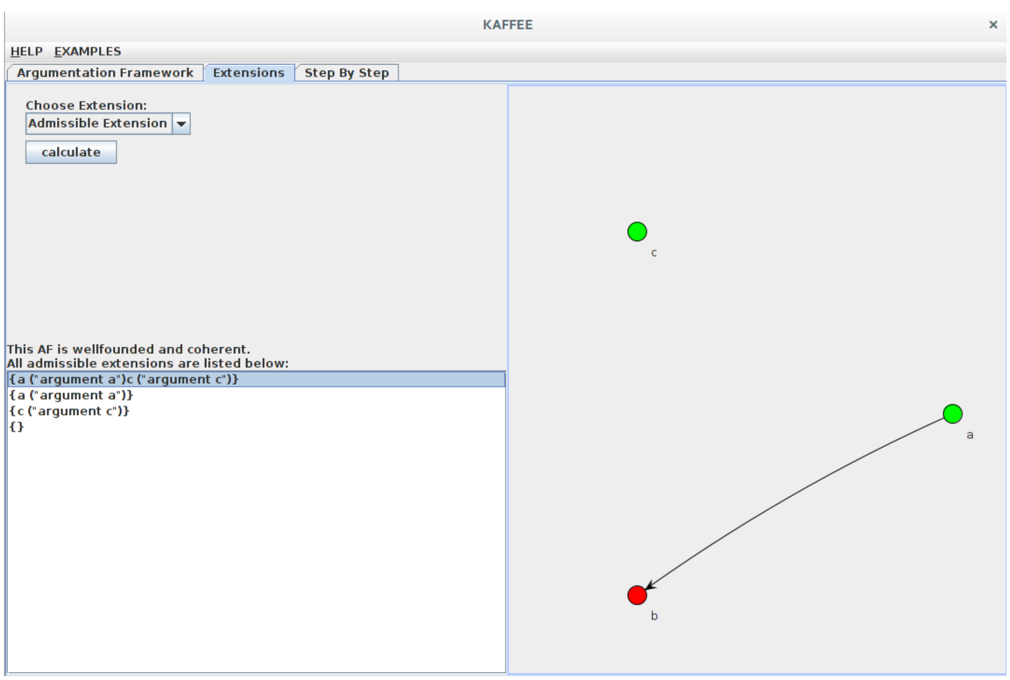

Figure 5.3: Tab "Extensions"

The last (and optional) tab is the "Step By Step" tab. Here, one can get a better explanation, in a walkthrough kind of style, of how admissible sets are created.

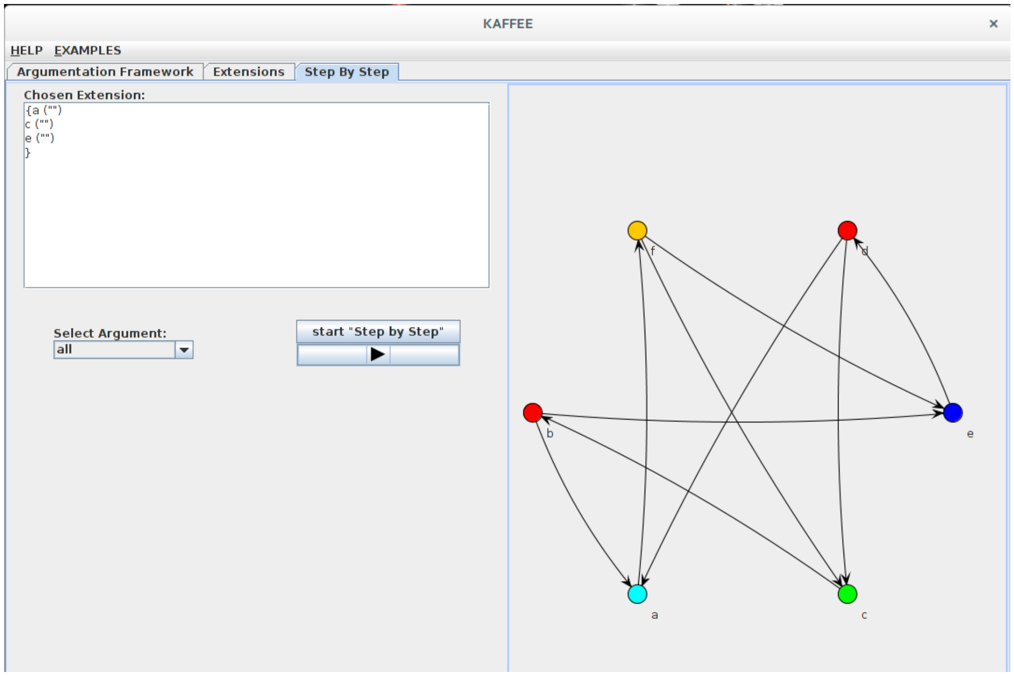

Figure 5.4: Tab "Step By Step"

There are also a few built-in shortcuts that make the usage of **KAFFEE** a little easier:

ALT+h . . . . Help menu

- **ALT+g** . . . link to the gibhub project containing all the sources, code etc
- **ALT+c** . . . feel free to email me

ALT+e . . . . Examples menu

- **ALT+2** . . . import example 1.4
- **ALT+3** . . . import example 1.5
- **ALT+4** . . . import example 1.6
- **ALT+s** . . . import example "Step By Step"

References

[1] P. M. Dung, "On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games," *Artificial Intelligence*, vol. 77, no. 2, pp. 321–357, 1995.

[2] U. Egly, S. A. Gaggl, and S. Woltran, "Answer-set programming encodings for argumentation frameworks," *Argument & Computation*, vol. 1, no. 2, pp. 147–177, 2010.

[3] M. Caminada, W. Carnielli, and P. Dunne, "Semi-stable semantics," *Journal of Logic and Computation*, vol. 22, no. 5, pp. 1207–1254, 2012.

[4] P. M. Dung, P. Mancarella, and F. Toni, "Computing ideal sceptical argumentation," *Artificial Intelligence*, vol. 171, no. 10-15, pp. 642–674, 2007.

[5] N. Gorogiannis and A. Hunter, "Instantiating Abstract Argumentation with Classical Logic Arguments: Postulates and Properties," *Artificial Intelligence*, vol. 175, no. 9-10, pp. 1479–1497, 2011.

[6] R. Reiter, "A logic for default reasoning," *Artificial Intelligence*, vol. 13, no. 1-2, pp. 81–132, 1980.

[7] P. Besnard and A. Hunter, *Elements of Argumentation*. MIT Press, 2008, ISBN: 978-0-262-02643-7.

[8] H. Prakken, "Some reflections on two current trends in formal argumentation," in *Logic Programs, Norms and Action - Essays in Honor of Marek J. Sergot on the Occasion of His 60th Birthday*, A. Artikis, R. Craven, N. K. Cicekli, B. Sadighi, and K. Stathis, Eds., ser. Lecture Notes in Computer Science, vol. 7360, Springer, 2012, pp. 249–272, ISBN: 978-3-642-29413-6.

[9] J. L. Pollock, "Defeasible Reasoning," *Cognitive Science*, vol. 11, no. 4, pp. 481–518, 1987.