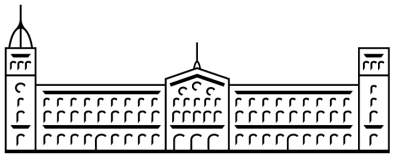

Final research project

GRAU DE MATEMÀTIQUES

Facultat de Matemàtiques
Universitat de Barcelona

Information fusion for norm consensus in virtual communities

Author: Marc Serramià Amorós

Supervisors: Dra. Maite López Sánchez
Dr. Juan Antonio Rodríguez Aguilar

Made at: Departament de matemàtica aplicada i anàlisi

Barcelona, February 12, 2016

Subject classification

68T99 (AMS 2010 Classification)

Keywords and phrases

Argumentation system, argumentation framework, information fusion, norm argument map, norm support, aggregation operators, social networks.

Abstract

Legislation in virtual communities should not be a private privilege, as well as the application of norms. In this project a new way of deliberating about norms is formalized, with the norm argument map, people will be able to argue and decide the norms that rule their virtual community. Also, we will see a method to fuse all the data introduced by participants into a single score or opinion about a given norm, from this the decision of wether to establish the norm or not will be taken. All of these decisions have to be taken in the most human way possible, to evaluate the results of the method, a Java project has been coded and by comparing it with other methods we will reason why this project's method is better.

Acknowledgements

Thanks to both of my directors Maite and Juan Antonio, their passion for the subject has been an inspiration and their help and guidance extremely valuable. Also thanks to my family and friends for always being there when I needed them.

Contents

1 Introduction **4**

1.1 Motivation 5

1.2 Goals 5

1.3 Document structure 5

2 Norm argument maps and aggregation operators **7**

2.1 Argumentation systems 7

2.2 Norm argument maps 8

2.3 Information fusion: Aggregation operators 10

3 Argument support **12**

3.1 The semantics of argument opinions 12

3.2 Argument support function 15

3.3 Opinion importance function construction and examples 16

3.3.1 Equitable opinion importance function examples 16

3.3.2 Positively biased opinion importance function examples 18

3.4 Opinion aggregation example 20

4 Argument set support **21**

4.1 $\boldsymbol{\alpha}$-Relevant arguments 21

4.2 Argument set support function 23

4.3 Argument importance function construction and alpha choice 25

5 Norm support **27**

5.1 Norm support function 27

5.1.1 Case $R_\alpha(A_n^+) \neq \emptyset$ and $R_\alpha(A_n^-) \neq \emptyset$ 28

5.1.2 Other cases 29

5.2 Detailed example of the overall norm support assessment process . . . 30

6 Particular Case: The social network **33**

6.1 Formalizing the problem . . . 33

6.2 Comparing aggregation methods . . . 34

6.2.1 Comparison with the naive average approach . . . 34

6.2.2 Discussion about the method in [5] . . . 37

7 Conclusions and future work **39**

7.1 Conclusions . . . 39

7.2 Future work . . . 40

Appendices **42**

A Norm argument map code and mathematica notebooks **43**

B Norm argument map code user guide **44**

B.1 The namargumentationsystem package . . . 44

B.2 The norm argument map file . . . 50

B.3 The NormSupportEvaluation Java project . . . 51

Chapter 1

Introduction

Norms are the obligations everyone has to respect to keep our world civilized, but for the norms to be fair, they must not be imposed by a dictator or an absolute monarch, they have to be argued by all the people that will be affected by them. Argumentation systems have been discussed at research level but, to the best of our knowledge have not been adopted by current virtual communities. In this research project, the work in [5] is continued in order to achieve a way of discussing norms in virtual communities and hence moderating without moderators. The core idea in this project, the norm argument map, is introduced in [5], but in that work is not sufficiently formalized. This project formally defines norm argument maps, along with other new concepts necessary to build a fair system.

This work also discusses about how to implement a way for the system to aggregate opinions of participants into the decision of wether to enact a norm or not. The naive average approach to this aggregation is not suitable, sometimes giving unexpected results. In order a system to be fair and easily adopted by humans, results need to be intuitive and provide aggregations that turn out to be as close as possible to what humans would decide by themselves, with a naive average approach this does not occur most of the time. We define a further elaborated approach which then will compare to the naive approach by means of some examples. The approach in [5] is not suitable either and we will discuss why.

Also, since the aim of this project is to make actual virtual communities more democratic, the theoretical work presented here is complemented with a Java implementation of the proposed approach. This implementation, together with the user guide, are both appended to the project and available on the web.

1.1 Motivation

In today's world, democracy is not only a system of governing a political entity, we should also see it as a requirement for a virtual community. The internet is omnipresent in our lives and growing every year with more users and virtual communities. These communities, not being perfect, sometimes need moderation and in the most democratic era of human history we should not give the privilege of controlling content and interpreting the community legislation to a few moderators, as to the best of our knowledge has been done in almost every community. Furthermore, moderators may have limitations for dealing with the task of moderating all the content and they can have biased opinions or even personal interests. So, to solve this problem, we need to give the power to the users, and to do so we require a way of discussing the norms in our social communities and deliberate about which one to apply when a conflict takes place.

1.2 Goals

The main goal of this final research project is to help develop a democratic way to discuss about norms in virtual communities and to define a way to aggregate people's opinions and arguments in order to reach consensus. So, particularly the goals of this research project are the following:

- To formalize the notion of norm argument map and describe its components.
- To establish a way to asses the support for an argument.
- To establish a way to asses the support for an argument set.
- To establish a way to asses the support for a norm.
- To compare the support assessment method described in this project with others.
- To implement a Java package with the structure of the norm argument map and its support functions along with a user interface to test it.

1.3 Document structure

This document describing the final research project has been structured as follows. Chapter 2, introduces the necessary background and a formalizes the definition for the norm argument map. Chapter 3, specifies the semantics of participants' opinions, assesses the support for an argument and provides some examples. In Chapter 4, arguments are sifted in order to get α-relevant arguments. Chapter 4

also explains how these arguments' supports are aggregated into a single argument set support. Chapter 5, introduces the assessment of a norm using argument set support. Chapter 6, discusses about how to apply the method to virtual communities (a social network case) and compares the method described in this project to others. Finally, in Chapter 7, the conclusions and future work are given. Readers can find in the appendix a reference where to find the code, which is implemented in Java. Along with the Java program, the reader can also find a user guide to use it and some mathematica notebooks used to find examples in Chapters 3 and 4.

Chapter 2

Norm argument maps and aggregation operators

In this chapter, some of the background notions are introduced, such as argumentation systems and aggregation operators. These ideas are explained in Sections 2.1 and 2.3. In Section 2.2 norm argument maps are formalized, since they have been used before, but there is a lack of a formal definition.

2.1 Argumentation systems

An argumentation system or argumentation framework is a structure designed for keeping a discussion with arguments. In [1], this structure is formally defined as a pair $AF = \langle \mathcal{A}, \rightharpoonup \rangle$, where $\mathcal{A}$ is a finite set of arguments and $\rightharpoonup \subseteq \mathcal{A} \times \mathcal{A}$ is a defeat relation. An argument a is said to defeat another argument b if $(a, b) \in \rightharpoonup$ (or sometimes written as $a \rightharpoonup b$). This argumentation framework can be represented as a directed graph in which vertices are arguments and directed arches characterize defeat among arguments, hence, a directed arch from vertex representing argument a to vertex representing argument b, means that a defeats b(in other words, a is a counter argument of b). To asses the acceptability of the arguments, one possible approach is to define argument labellings, an argument can be accepted(by labelling it as `in`), rejected(by labelling it as `out`) or undecided wether to accept or reject it(by labelling it as `undec`). Formally this argument labelling is a total function $L : \mathcal{A} \to \{\texttt{in}, \texttt{out}, \texttt{undec}\}$. Note that, a labelling should be conflict-free(that is, $\forall a, b \in \texttt{in}(L), \neg(a \rightharpoonup b)$, where $\texttt{in}(L)$ is the set of arguments that are labelled `in`). From these definitions different labelling semantics are introduced, such as: complete labelling, grounded labelling, preferred labelling, stable labelling, etc.

In [1] the argumentation framework is formally defined, but in practice argumentation frameworks can have a different structure, such as the one defined in [3]. In that work, the author introduces another approach, which is an argumentation system

that serves as a way to have a discussion on a large scale, keeping it summarized, with the goal of reaching consensus. This structure, called argument map, is not formalized in [3], but the author defines it as a tree with a deliberation question as the root and arguments as children, each child argument has its own children hence creating argumentation branches. Participants of the debating process can add arguments to the tree, these arguments are checked by moderators and relocated in the tree if necessary.

2.2 Norm argument maps

In [5] the author explains another argumentation system, the norm argument map, which is the system studied in this research project. The norm argument map is an argumentation system with the goal of discussing a norm. The major difference between the norm argument maps and the argument maps is that, in the norm argument maps, the discussion happens in a more restricted space, in the sense that the argumentation is not structured in argumentation branches but in two sets of supporting and attacking arguments. This argumentation system was introduced in order to create norms in social networks, a norm is an established expected behavior, for example, in the case of a social network, a norm could be: *"It is prohibited to post spam in the multimedia section of the social network"*.

The goal of the system explained in [5] is to avoid conflicts. Conflicts are undesired states of the system, posts that visitors complain about in a sufficiently big proportion, for example, a social network with plenty of insulting posts participants complain about. Conflicts can be detected when the ratio between visits and complaints exceeds from a predefined threshold. So, if a conflict is found, a deliberation is started, participants can then suggest a norm to solve the conflict, hence a norm argument map is automatically created so that all participants in the social community can debate about the norm. In that work, the author explains and structures this argumentation system but does not define it formally.

The goal of this research project is to provide the means to decide whether a norm should be accepted or not, hence a formal definition of norm argument map is needed. First we have to consider the definition of norm and argument:

Definition 2.2.1. *A **norm** is an object of the form $n = (\phi, \theta(\alpha))$, where ϕ is the norm's precondition, θ is a deontic operator and α is an action that can be performed by participants.*

The precondition is the condition that has to occur for the norm to be applied, the action is the participant's action that triggers the actual application of the norm and for the deontic operator see [4].

Example. *As previously said, consider a social network with a multimedia section where people could post multimedia about their hobbies, then this network could have*

the norm "Do not post spam in the multimedia section". In this case the precondition would be "be in the multimedia section", the deontic operator would be "prohibition", and the action could be "upload spam content". So, if a person posted spam in the multimedia section the norm would apply and that post would be labelled as prohibited.

To discuss a norm, participants will use arguments and opinions. Arguments, in general, are statements that participants assign to a norm and that represent either a support for the norm or an attack to it. Opinions are valuations that participants assign to arguments, in the case of the norm argument map, opinions will be in the form of numerical values that people will provide for other people's arguments, the value will quantify the extent to which they support the argument or they are against it. These opinions, in the case of the norm argument maps, are not complete, so participants can give an opinion for a given argument but they are not forced to.

Definition 2.2.2. *An **argument** in a norm argument map is a pair $a_i = (s, \vec{O}_{a_i})$ of a statement s, the argument itself, and a vector of opinions $\vec{O}_{a_i}$ which contains all the opinion values assigned to the argument and provided by the participants.*

Notation.

$\vec{O}_{a_i} = (o^i_1, \ldots, o^i_{n_i})$ is the vector of opinions, note that o^i_j is the j^{th} opinion about argument a_i, and that since each argument a_i has different number of opinions (n_i), the last opinion about argument a_i is the opinion $o^i_{n_i}$.

Determining whether two given arguments are in fact the same is out of the scope of this work, and thus, we will consider arguments to be different. Therefore it becomes natural to consider the idea of a set of arguments or argument set:

Definition 2.2.3. *An **argument set** is a collection of arguments. An argument set can be defined as such or, more practically, it can be defined related to a norm. Thus, a **norm argument set** is an argument set related to a norm n, so arguments in the norm argument set are arguments supporting or attacking the norm n.*

Notation.

Argument set: $A = \{a_1, \ldots, a_k\}$, with a_i being arguments for $i \in \{1, \ldots, k\}$.

Norm argument set related to a norm n: $A_n = \{a_1, \ldots, a_j\}$, with a_i being arguments for $i \in \{1, \ldots, j\}$.

$\vec{O}_A$ is the vector of all the opinions of the arguments in A.

With these basic concepts a norm argument map can be defined as follows:

Definition 2.2.4. *A **norm argument map** $M = (n, A_n, \kappa)$ is a triple of a norm n, a norm argument set A_n and a function κ that classifies the arguments of A_n between the ones that are in favor of the norm and the ones that are against it.*

Comment. *For convenience, we will refer hereafter to the two different argument sets that result from the partition of κ over A instead of the norm argument set itself. Specifically, we will denote the set of arguments in favor of norm n as A_n^+, and the set of arguments against the norm as A_n^-.*

For the sake of simplicity, we will also refer to the set of arguments in favor of a norm as (the set of) positive arguments, whereas we will refer to the set of arguments against a norm as (the set of) negative arguments.

In a norm argument map participants can add arguments to A_n, and can also give their opinions on other people's arguments. Consequently if $a_i = (s, \vec{O}_{a_i})$ is an argument with $\vec{O}_{a_i} = (o_1^i, \ldots, o_j^i)$ and a participant gives his/her opinion about the argument by assigning it a value v, now the argument a_i becomes $a_i' = (s, \vec{O}_{a_i'})$ with $\vec{O}_{a_i'} = (o_1^i, \ldots, o_j^i, v)$.

Thus, whenever a conflict arises in the social network, participants have the opportunity to discuss whether a specific norm can solve this conflict. But in general, different norms can be discussed at once, so we consider these discussions take place in a framework defined as follows:

Definition 2.2.5. *A **norm argument map framework** $F = (P, N)$ is a pair of a set of participants P and a set of norm argument maps N, so that participants in P can deliberate about different norms by means of the norm argument maps in N.*

2.3 Information fusion: Aggregation operators

In order to give an overall support to a norm, it is necessary to aggregate all the different opinions and supports in the norm argument map, hence aggregation operators are needed.

In this project some of the operators presented in [11] are used, their definition can be found below along with other definitions of some additional concepts that are necessary for these aggregations.

Definition 2.3.1. *A **weighting vector** w is a vector such that if $w = (w_1, \ldots, w_n) \in \mathbb{R}^n$ then $w_i \in [0, 1]$ and $\sum_{i=1}^n w_i = 1$.*

If we have an element vector $e = (e_1, \ldots, e_n) \in \mathbb{R}^n$ and we aim to aggregate the e_i's, we can define different ways of aggregating them. If we want to weigh the position inside the vector e to aggregate them, in other words, weigh the importance of the information source that gave that element, we could use a weighted mean:

Definition 2.3.2. *Let $w = (w_1, \ldots, w_n) \in \mathbb{R}^n$ be a weighting vector and let $e = (e_1, \ldots, e_n) \in \mathbb{R}^n$ be the vector of elements we want to aggregate. A **weighted mean** is a function $WM_w(e) : \mathbb{R}^n \to \mathbb{R}$, defined as $WM_w(e) = \sum_{i=1}^n w_i e_i$.*

Note that ,with the weighted mean, we can assign more or less importance to a particular information source, being each position in the e vector our information sources. If we want to weigh the values of e themselves and not its positions in the vector we have to use an OWA, this operator was introduced by Yagger in [12] and it is defined as follows:

Definition 2.3.3. *Let $q = (q_1, \ldots, q_n) \in \mathbb{R}^n$ be a weighting vector and let $e = (e_1, \ldots, e_n) \in \mathbb{R}^n$ be the vector of elements we want to aggregate. An **ordered weighted average or OWA** is a function $OWA_q(e) : \mathbb{R}^n \to \mathbb{R}$, defined as $OWA_q(e) = \sum_{i=1}^{n} q_i e_{\sigma(i)}$, where σ is a permutation of the e_i so that they are ordered from the largest one to the lowest one, hence $e_{\sigma(i)}$ is the i^{th} largest number in e.*

Thus, as previously stated, if we want to give more importance to high/central/low values, we would use a weighting vector q with higher values at the start/center/end.

If we want to weigh both the information source and the value of the numbers aggregated, we have to use a WOWA operator. This operator was introduced by V. Torra in [7]. In fact, it can be defined in two different ways but the one used in this research project corresponds to the definition using vectors given in [7]:

Definition 2.3.4. *Let $w = (w_1, \ldots, w_n) \in \mathbb{R}^n$ and $q = (q_1, \ldots, q_n) \in \mathbb{R}^n$ be two weighting vectors and let $e = (e_1, \ldots, e_n) \in \mathbb{R}^n$ be the vector of elements we want to aggregate. A **weighted ordered weighted average, weighted OWA or WOWA** is a function $WOWA_{w,q}(e) : \mathbb{R}^n \to \mathbb{R}$ defined as:*

$$WOWA_{w,q}(e) = \sum_{i=1}^{n} p_i e_{\sigma(i)}$$

Where σ is a permutation of the elements in e so that $e_{\sigma(i)}$ is the i^{th} largest element in e and:

$$p_i = f^*(\sum_{j \leq i} w_{\sigma(j)}) - f^*(\sum_{j<i} w_{\sigma(j)})$$

Where f^* is a non decreasing interpolation function of the points:

$$\{(i/n, \sum_{j \leq i} q_j)\}_{i=1,\ldots,n} \cup \{(0,0)\}$$

This function has to be a straight line when the points can be interpolated that way.

Comment. *Note that, w acts as the vector in the weighted mean, weighting the information source, while q acts like the OWA vector, weighting the value of the elements being aggregated.*

Chapter 3

Argument support

In order to compute the global support for a norm, we first need to compute the support for each one of its arguments. Arguments are the only part of the norm argument map that participants can create and give opinions about, thus, the procedure will establish the support for the debated norm by considering its arguments and their associated opinions.

3.1 The semantics of argument opinions

In order to compute a meaningful aggregation of argument opinions, it becomes necessary to define their associated semantics. Although different semantics can be associated to participants' opinion values, this section proposes one that fits well the purposes of norm discussion. Needless to say, alternative semantics could also be studied, but they remain as future work. In what follows, several definitions related to the proposed semantics are presented.

Definition 3.1.1. *An **opinion spectrum** or, simply, **spectrum** is the set λ of possible numerical values individual participants can assign to each argument meaning his/her opinion about that argument.*

The spectrum λ will be the domain of functions operating with opinion values, such as the importance function defined below. Hence, the same spectrum will be considered for all arguments in the argumentation system.

The spectrum could be a subset of $\mathbb{Z}$ or $\mathbb{N}$, but I will be considering a more general case, the case $\lambda \subset \mathbb{R}$. Note that in this case λ has to be a connected, closed and bounded subset of $\mathbb{R}$ (i.e. a closed interval), it has to be closed and bounded so that participants can give maximum and minimum opinions and it has to be connected for the opinions to have a continuity of meaning in the spectrum. So, from now on, we characterize λ as $[lb, ub]$ where lb and ub correspond respectively to the lower bound and the upper bound of the spectrum.

Since the participants have to give opinions about arguments, we have to choose one of the possible semantics. The semantic chosen in this research project is that the bigger end of the spectrum means to assign an opinion in favor of the argument and symmetrically, the lower end to assign an opinion that goes against it. Therefore, the opinion ub is the strongest opinion in favor of an argument, whereas lb is the strongest opinion against it. The opinion lying in the middle of the interval $\frac{lb+ub}{2}$ has a non-positioned meaning, a neutral opinion, so we do not credit it any importance. This way of describing the meaning of opinions separates the spectrum in three different meaning blocks, the against block $[lb, \frac{lb+ub}{2})$, the neutral block $\{\frac{lb+ub}{2}\}$ and the pro block $(\frac{lb+ub}{2}, ub]$. The ub and lb are the most important opinions of the pro and against blocks respectively, the opinions in between are not as strong as them so they are not as important either. Figure 3.1 shows an example of the previously stated in the spectrum $[1, 5]$.

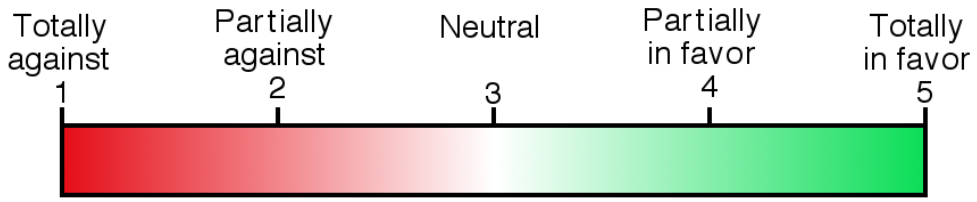

Figure 3.1: The meaning of opinions in the spectrum $\lambda = [1, 5]$ with some representative opinion values' meanings.

Since a meaning has to be assigned to each number in $\lambda = [lb, ub]$, an importance function $I(x)$ has to be defined to weight the importance of each argument opinion reflecting the considerations previously stated. This $I(x)$ function will be different depending on the problem we are facing (possible layouts are described in Section 3.3).

Definition 3.1.2. *Let $\lambda = [lb, ub]$ be a spectrum, an **importance function** is a function $I(x) : \lambda \to [0, 1]$ that for each argument opinion in λ assigns its importance in $[0, 1]$, satisfying the next conditions:*

- *(C1) $I(x)$ has to be a continuous and piecewise differentiable function*
- *(C2) $I(ub) = 1$*
- *(C3) $I(\frac{lb+ub}{2}) = 0$*
- *(C4)* $\begin{cases} I'(x) < 0 & \text{if } x \in [lb, \frac{lb+ub}{2}) \text{ and } I \text{ is differentiable in } x \\ I'(x) = 0 & \text{if } x = \frac{lb+ub}{2} \text{ and } I \text{ is differentiable in } x \\ I'(x) > 0 & \text{if } x \in (\frac{lb+ub}{2}, ub] \text{ and } I \text{ is differentiable in } x \end{cases}$

Comment. *The function has to be continuous and piecewise differentiable in order to be able to satisfy condition C4. Since the highest value in $I(x)$'s codomain is 1, $I(ub)$ has to be 1, because as we have stated, ub is the strongest opinion in the pro block (the value of $I(lb)$ is treated in definitions 3.1.3 and 3.1.4). $I(\frac{lb+ub}{2})$ has to be zero because we have established that $\frac{lb+ub}{2}$ is the less important opinion, in fact we have considered it negligible, so we assign it the minimum number in the function's codomain. The function has to be decreasing (i.e., $I'(x) < 0$) in the subinterval $[lb, \frac{lb+ub}{2})$ because lb is the most important opinion of this interval (the against block) and the nearer an opinion is to $\frac{lb+ub}{2}$ the less important is, analogously the function has to be increasing (i.e., $I'(x) > 0$) in the subinterval $(\frac{lb+ub}{2}, ub]$, and if the function satisfies these two conditions then the only possible value for $I'(\frac{lb+ub}{2})$ is 0, if I is differentiable in that point.*

We can consider different importance functions to be opinion importance functions $I_{opinion}(x)$. In this research project I have defined two types that I consider suit the most the semantic previously defined. These two opinion importance functions are defined as follows:

Definition 3.1.3. *An importance function $I_{opinion}$ is an **equitable opinion importance function** if it satisfies that $I_{opinion}(\frac{lb+ub}{2} + x) = I_{opinion}(\frac{lb+ub}{2} - x)$ for $x \in [0, \frac{lub-lb}{2}]$.*

Note that, this type of function gives the same importance to the pro and against block, and since it is a symmetric function, it is fair, but it is prone to suffer from a group of people systematically giving the lowest opinion value to an argument. If that happens, the overall support can be not adequate in the sense that it is not the support that should have, therefore the argument will not be as important as it should be in the overall process. To solve this problem a new type of function can be defined:

Definition 3.1.4. *An importance function $I_{opinion}(x)$ is a **positively biased opinion importance function** if it satisfies that $I_{opinion}(lb) < I_{opinion}(ub)$.*

With a positively biased opinion importance function we are giving more importance to positive opinions, hence if a group of people systematically gives low value opinions to an argument, unless it is a huge group, the argument will have a more balanced support. This type could be used for discussing controversial norms or used in environments with high numbers of participants intentionally trying to affect the support for a norm. This biased treatment is actually balanced out by itself, because both the pro arguments' opinions and the con arguments' opinions are exposed to this function, so if someone really wants to impact negatively to the norm it is more effective to include a good con argument than not reasoning and only giving lower value opinions to pro arguments, and analogously if someone wants a norm to be established it is more effective to leave a good pro argument than giving low value opinions in the con arguments.

This function can be defined reducing the importance of all the against block opinions or, since people trying to affect negatively the support might only give opinions on the most extreme contrary value, you can have a function that is symmetrical except for the extreme against end, for example, if $\lambda = [lb, ub]$, then you could define $I_{opinion}(x)$ as an equitable importance function in $[lb + k, ub - k]$, $k \in (0, \frac{ub-lb}{2}]$ and then define the function with higher importance in the $(ub - k, ub]$ interval than in the $[lb, lb + k)$. By only reducing the importance in the most extreme against opinions we have a positively biased opinion importance function that only affects extreme opinions, the ones suspicious of being used to affect negatively, and not the more moderate ones that might be awarded by normal participants.

Comment. *Note that, the same semantic that has been applied to opinions can be applied to an argument support, an argument set support or a norm support once we know how to asses them.*

3.2 Argument support function

The goal of this section is to define a function that takes an argument and assesses its support within the same spectrum as the argument. We have a function that weighs each opinion's importance, now we have to define a support function using an aggregation operator to aggregate the opinions considering their importance.

Let $a_i \in A$ be an argument and let $\vec{O}_{a_i} = (o^i_1, \ldots, o^i_{n_i}), o^i_j \in \lambda = [lb, ub]$, be its associated opinions. Each o^i_j is a real number in $[lb, ub]$ and has its own importance given by $I_{opinion}(o^i_j)$. In Chapter 2, we have introduced two operators suitable for this kind of aggregation, the weighted mean (WM_w) and the OWA. Since we want to aggregate numbers weighting their values an OWA should be considered, but in this case we will consider a weighted mean with a function giving the weights, the opinion importance function. The reason to do so is explained in subsequent Section 3.4.

Definition 3.2.1. *An **argument support function** is a function $S_{arg} : A \to \lambda$, where A is the set of arguments and $\lambda = [lb, ub]$ the spectrum, that for each argument $a_i \in A$, having n_i associated opinions, computes its support as follows:*

$$S_{arg}(a_i) = WM_w(\vec{O}_{a_i})$$

using the weighting vector:

$$w = \Big(\frac{I_{opinion}(o^i_1)}{T}, \ldots, \frac{I_{opinion}(o^i_{n_i})}{T}\Big), \text{ where } o^i_j \in \vec{O}_{a_i},\ j \in \{1, \ldots, n_i\}$$

$$\text{and } T = \sum_{j=1}^{n_i} I_{opinion}(o^i_j)$$

Being o_j^i the j^{th} opinion of argument a_i and T the overall addition of all importance values associated to all opinions about argument a_i, so that the elements in w sum one and therefore w is a weighting vector.

3.3 Opinion importance function construction and examples

The opinion importance function depends on the problem: for different deliberations settings one has to consider different functions, thus, the way this function is constructed is not fixed. The example functions in this research project have been constructed with two different methods: by interpolation and geometrically. The first one, is constructed by defining some target points (those that are desired to be in the function) and then interpolating these points. In this case, one might have to interpolate numerous parts, and that is the reason for the importance function to be required to be only piecewise differentiable. Nevertheless, having sufficient points and sufficient parts the desired opinion importance function can be constructed.

These are some possible opinion importance functions for the described layout in each case, Mathematica notebooks with the implementation are referenced in the appendix:

3.3.1 Equitable opinion importance function examples

The equitable natural opinion importance function

This function has concave endings and convex neutral middle part, hence opinions in the extremes of the blocks have similar importance while in the middle of each block the change is significant, this might be the most natural way to implement the opinion importance function, it has been divided into 5 pieces each one has been found interpolating a set of points as follows:

f_1: Interpolation of $\{(lb, 1), (\frac{ub+7lb}{8}, 0.9), (\frac{ub+3lb}{4}, 0.65)\}$:

$$f_1(x) = \frac{ub^2 - 1.8\ ub\ lb - 0.2\ ub\ x - 4\ lb^2 + 9.8\ lb\ x - 4.8\ x^2}{(lb - ub)^2} \text{ when } x \in \left[lb, \frac{ub + 3lb}{4}\right]$$

f_2: Interpolation of $\{(\frac{ub+3lb}{4}, 0.65), (\frac{3ub+5lb}{8}, 0.25)\}$:

$$f_2(x) = \frac{1.45\ ub + 1.75\ lb - 3.2\ x}{ub - lb} \text{ when } x \in \left(\frac{ub + 3lb}{6}, \frac{3ub + 5lb}{8}\right)$$

f_3: Interpolation of $\{(\frac{3ub+5lb}{8}, 0.25), (\frac{ub+lb}{2}, 0), (\frac{5ub+3lb}{8}, 0.25)\}$:

$$f_3(x) = \frac{4\ ub^2 + 8\ ub\ lb - 16\ ub\ x + 4\ lb^2 - 16\ lb\ x + 16\ x^2}{(lb - ub)^2} \text{ when } x \in \left[\frac{3ub + 5lb}{8}, \frac{5ub + 3lb}{8}\right]$$

f_4: Interpolation of $\{(\frac{5ub+3lb}{8}, 0.25), (\frac{3ub+lb}{4}, 0.65)\}$:

$$f_4(x) = \frac{1.75\ ub + 1.45\ lb - 3.2\ x}{lb - ub} \text{ when } x \in \left(\frac{5ub + 3lb}{8}, \frac{3ub + lb}{4}\right)$$

f_5: Interpolation of $\{(\frac{3ub+lb}{4}, 0.65), (\frac{7ub+lb}{8}, 0.65), (ub, 1)\}$:

$$f_5(x) = \frac{-4\ ub^2 - 1.8\ ub\ lb + 9.8\ ub\ x + lb^2 - 0.2\ lb\ x - 4.8\ x^2}{(lb - ub)^2} \text{ when } x \in \left[\frac{3ub + lb}{4}, ub\right]$$

Figure 3.2: Example of an equitable natural opinion importance function being $\lambda = [1, 5]$.

Figure 3.2 shows an example of this function in the spectrum $\lambda = [1, 5]$, so that that f_1 is defined in $[1, 2]$, f_2 in $(2, 2.5)$, f_3 in $[2.5, 3.5]$, f_4 in $(3.5, 4)$ and f_5 in $[4, 5]$.

The equitable extremist opinion importance function

In this case, the proposed function is a convex parabola, hence in this case extrem opinions are the most important ones by difference. This importance function may be suitable for cases when clear positions are needed. If the norm debated will change radically a lot of aspects of the community then it can be announced that this importance is going to be applied to the opinions, hence people that want to influence (that is, that their opinion is really taken into account) will have to position themselves clearly. Figure 3.3 shows an example of this function in the spectrum $\lambda = [1, 5]$.

To find this function , it would be possible to interpolate three points, the two lying on the extremes and the vertex of the parabola, but in fact, since we know that the function is a parabola, we can thus reason as follows:

A parabola with the vertex at the point $(\frac{lb+ub}{2}, 0)$ is going to be of the form : $f(x) = (x - \frac{lb+ub}{2})^2$. Since we want the maximum value in the interval $[lb, ub]$ to be 1 we have to divide by the value of the parabola in one of the extremes, hence:

$$f(x) = \frac{(x - \frac{lb+ub}{2})^2}{(lb - \frac{lb+ub}{2})^2} = \frac{(lb + ub - 2x)^2}{(lb - ub)^2} \text{ when } x \in \left[lb, ub\right]$$

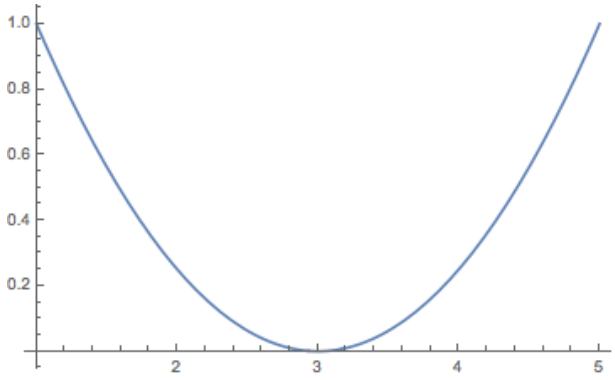

Figure 3.3: Example of an equitable extremist opinion importance function being $\lambda = [1, 5]$.

3.3.2 Positively biased opinion importance function examples

The positively biased natural opinion importance function

The previous section defines the natural opinion importance function in its equitable version. To consider the positively biased version we only have to change f_1:

f_1: Interpolation of $\{(lb, 0.84), (\frac{ub+7lb}{8}, 0.8), (\frac{ub+3lb}{4}, 0.65)\}$:

$$f_1(x) = \frac{-2.56\ lb^2 - 1.8\ lb\ ub + 6.92\ lb\ x + 0.84\ ub^2 + 0.12\ ub\ x - 3.52\ x^2}{(lb - ub)^2}$$

$$\text{when } x \in \left[lb, \frac{ub + 3lb}{4}\right]$$

Figure 3.4 shows an example of the positively biased natural opinion importance function in the spectrum $\lambda = [1, 5]$, note that the lower ending of the function has smaller values than the bigger ending.

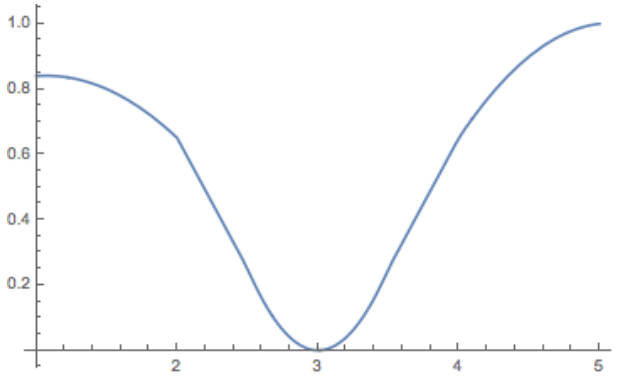

Figure 3.4: Example of a positively biased natural opinion importance function being $\lambda = [1, 5]$.

The positively biased extremist opinion importance function

The previous section defines the equitable version of the extremist opinion importance function. To construct the positively biased version we could consider the function in two pieces f_1 and f_2 as follows:

$$f_1(x) = \frac{0.8\ (x - \frac{lb+ub}{2})^2}{(lb - \frac{lb+ub}{2})^2} = \frac{0.8\ (lb + ub - 2x)^2}{(lb - ub)^2} \text{ when } x \in \left[lb, \frac{ub + lb}{2}\right]$$

$$f_2(x) = \frac{(x - \frac{lb+ub}{2})^2}{(lb - \frac{lb+ub}{2})^2} = \frac{(lb + ub - 2x)^2}{(lb - ub)^2} \text{ when } x \in \left[\frac{lb + ub}{2}, ub\right]$$

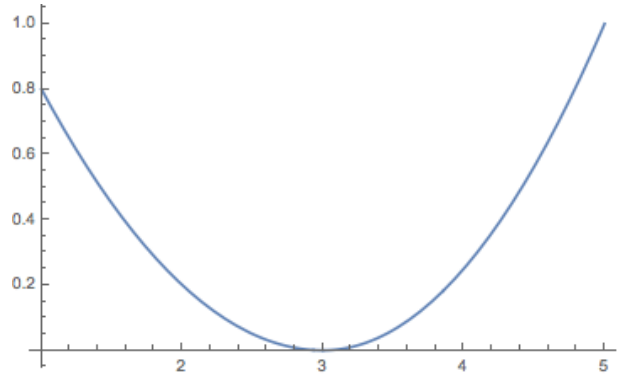

Figure 3.5: Example of a positively biased extremist opinion importance function being $\lambda = [1, 5]$.

Figure 3.5 shows an example of the positively biased extremist opinion importance function in the spectrum $\lambda = [1, 5]$, as in Figure 3.4, the lower ending of the function has smaller values than the bigger ending.

3.4 Opinion aggregation example

Consider a norm n with the spectum $\lambda = [1, 5]$. Suppose we have an argument a_1 with opinions $\vec{O}_{a_1} = (5, 5, 5, 4, 4, 4, 3, 3, 3)$. Using the support function defined previously and considering the equitable natural opinion importance function, the resulting support is: $S_{arg}(a_1) = 4,6061$, this is accurate, because there are 6 opinions in favor (3 of them extremely favorable) and 3 other opinions that are neutral and we previously decided to discard.

The reason to avoid using an OWA is that, for example, if we wanted to solve the same example, we should define a weighting vector representing the semantic given to opinions, since our semantic gives more importance to extreme values a possible vector would be: $(0, 18, 0, 16, 0, 12, 0, 04, 0, 0, 04, 0, 12, 0, 16, 0, 18)$. This vector weighs the extreme numbers, as it should be, but since a_1 has opinions in only a subinterval of the spectrum, the use of the OWA distorts the result, in this case using the OWA the support would be 4. Since we consider opinions with value 3 to be negligible this result is not accurate.

Chapter 4

Argument set support

Since we now know how to calculate the support for each argument individually, we have to face the problem of calculating the support for a set of arguments. In the norm argument map, which was introduced in [5] and has been formally defined in Chapter 2, the sets of arguments that we have to asses the support for are A_n^+, the set of positive arguments, the ones that support the norm, and A_n^-, the set of negative arguments, the ones that attack the norm. It is clear that if we propose a way to compute the support for a general set of arguments A, in particular, the same procedure will also be suitable for computing the support for both A_n^+ and A_n^-.

4.1 α-Relevant arguments

Before we start defining the procedure to asses the support for an argument set, let us consider the following example:

Example. *There is a norm n with positive and negative arguments with the spectrum $\lambda = [1, 5]$. In particular, n has been argued with three positive arguments a_1, a_2, a_3 and a single negative argument $\overline{a_1}$. On the one hand, in the set of positive arguments a_1 has a support of 5, which comes from a single opinion while a_2 has a support of 1, which comes from 100 opinions and finally, a_3 has also a support of 1 coming from 100 opinions. On the other hand, on the set of negative arguments $\overline{a_1}$ has a support of 5, which comes from 30 opinions:*

A_n^+	$S_{arg}(a_i)$	$dim(\vec{O}_{a_i})$
a_1	5	1
a_2	1	100
a_3	1	100

A_n^-	$S_{arg}(\overline{a_i})$	$dim(\vec{O}_{\overline{a_i}})$
$\overline{a_1}$	5	30

What should we consider to give the support for A_n^+ on this extreme case? We should discard a_2 or a_3 because they have bad support, in fact the minimum possible support, hence people have decided that these arguments are not appropriate

or are not a valid reason to apply the norm under discussion. In Section 3.1 we have defined the meaning of an argument opinion value in $[lb, \frac{lb+ub}{2})$ as a contrary opinion to the argument. Afterwards, in a comment, the meaning of opinions is extrapolated and applied to all the supports. For this reason, arguments with supports outside of $(\frac{lb+ub}{2}, ub]$ are not accepted and, therefore, should not be considered as valid arguments.

We cannot consider a_1 either because although it has the maximum possible support it has only been validated by one person, hence it is negligible in front of the other arguments. One way to treat this, is to require a minimum percentage of opinions. Nevertheless, this option is not suitable sometimes because in case a lot of different arguments are posted with equal number of opinions the percentage would be minuscule and all arguments would be discarded. Alternatively, we can filter out arguments by just considering those having at least a number of opinions that corresponds to a certain fraction of the number of opinions of the argument with most opinions.

Thus, we tackle this argument relevance problem by creating a new subset of arguments containing only the arguments considered α-relevant and defining the criteria needed to be considered as such:

Definition 4.1.1. *Let A be a set of arguments with spectrum $\lambda = [lb, ub]$, a **relevant argument** of A is an $a_i \in A$ that satisfies:*

$$S_{arg}(a_i) > \frac{lb + ub}{2}$$

Definition 4.1.2. *Let A be a set of arguments with spectrum $\lambda = [lb, ub]$, and suppose $\alpha \in [0, 1]$ and $a_k \in A$ the argument with most opinions, an **α-relevant argument** of A is a relevant argument $a_i \in A$ that also satisfies:*

$$dim(\vec{O}_{a_i}) \geq \alpha \; dim(\vec{O}_{a_k})$$

Comment. *Note that, in the case of the norm argument map, the α-relevant arguments would be the ones that are α-relevant in A_n, that is, the ones that are relevant and that have $dim(\vec{O}_{a_i}) \geq \alpha \; dim(\vec{O}_m)$, being m the argument with most opinions in $A_n = A_n^+ \cup A_n^-$.*

Definition 4.1.3. *Let A be an argument set. The subset $\boldsymbol{R_\alpha(A)} \subseteq A$ is the subset containing all the α-relevant arguments of A. In the case of the norm argument map, $\boldsymbol{R_\alpha(A_n^+)}$ are the positive α-relevant arguments of the norm argument map, and analogously, $\boldsymbol{R_\alpha(A_n^-)}$ are the negative α-relevant arguments of the norm argument map.*

This new subset of arguments contains the arguments that have been considered α-relevant for the reasons previously explained, hence are the ones which its support should be aggregated to assign the support to the set of arguments A.

Once α-relevant arguments have been defined, we are now able to choose which arguments to consider in the previous example, we will consider $\alpha = 0,3$. In the case of the positive argument set, we have $R_{0,3}(A_n^+) = \emptyset$, because as explained before a_1 has not enough opinions and a_2 and a_3 have bad support. Nevertheless, in the case of the negative argument set, we have $R_{0,3}(A_n^-) = \{\overline{a_1}\}$, because $S_{arg}(\overline{a_1}) = 5 > 3$ and $dim(\vec{O}_{\overline{a_1}}) = 30 \geq 0,3 \cdot dim(\vec{O}_{a_2}) = 0,3 \cdot 100 = 30$.

Comment. *Note that, the need to have a minimum number of opinions for an argument to be considered α-relevant poses another problem. We have considered all arguments to be different. That being said, a set could be formed of hundreds of similar arguments with few opinions, that individually would not be considered α-relevant, but altogether sum a large quantity of opinions and represent a strong argument. Hence, similar arguments should be merged. This problem, which is out of the scope of this final research project, could be solved with natural language.*

4.2 Argument set support function

Let A be a finite set of arguments of a given norm n, with $R_\alpha(A) = \{a_1, \ldots, a_k\}$ and suppose that we know both: (i) $S_{arg}(a_i)\ \forall a_i \in R_\alpha(A)$, the support for each argument in $R_\alpha(A)$ and (ii) $\vec{O}_{a_i}\ \forall a_i \in R_\alpha(A)$, the opinions that each argument in $R_\alpha(A)$ has received. Then, we want to define a function $S_{set}(A) = S_{set}(R_\alpha(A))$ with values in $[lb, ub]$, the support function of a given set of arguments. To define S_{set} we have to consider different aspects of the nature of this aggregation.

Since participants will have given opinions of different arguments and some will have received more important opinions than others(that is., the overall importance of the opinions received may vary from argument to argument), it is natural to consider that the arguments with more important opinions should have more relevance to the support for the set. In addition, the arguments with higher support should have more impact to the support for the set too, because they have been widely accepted by participants as powerful arguments. In other words, we have to consider the importance of the information source, as well as, the values of the support, therefore, the WOWA operator suits well this kind of aggregation.

To define a WOWA operator we need two vectors, w the vector weighting the information source and q, the vector weighting the aggregation values. First, we define w: the more important opinions an argument has, the bigger its weight has to be. The weight of each argument should be the sum of the importance of its opinions, then each element in th vector should be divided by the appropriate number so altogether the elements sum one (required to be a weighting vector), hence

if $R_\alpha(A) = \{a_1, \ldots, a_{k'}\}$:

$$w = \Big(\frac{\sum_{j=1}^{dim(\vec{O}_{a_1})} I_{opinion}(o_j^1)}{\tau}, \ldots, \frac{\sum_{j=1}^{dim(\vec{O}_{a_{k'}})} I_{opinion}(o_j^{k'})}{\tau}\Big)$$

$$\text{with } \tau = \sum_{i=1}^{k'} \Big(\sum_{j=1}^{dim(\vec{O}_{a_i})} I_{opinion}(o_j^i)\Big) \text{ with } o_j^i \in \vec{O}_{a_i} = \{o_1^i, \ldots o_{n_i}^i\}$$

Comment. *Given the argument a_i, note that, $\sum_{j=1}^{dim(\vec{O}_{a_i})} I_{opinion}(o_j^i)$ represents the overall importance of the argument, since it is the sum of the importance of its opinions.*

Now, we define q, this vector is a weighting vector that has to weigh the importance of each argument's support. The higher the support the higher the weight in the vector. For $a_i \in R_\alpha(A)$, $S_{arg}(a_i)$ is a real number in $(\frac{lb+ub}{2}, ub]$, so we have to define its weight with a new function, as previously done to weigh the importance of opinions.

This new function has to satisfy some criteria. We consider the codomain $[0, 1]$, in this case, the weight of the maximum support (ub) should be 1. It is expected that the higher the support for the argument the higher weight should have. Hence, it has to be a strictly increasing function. To be a strictly increasing function we require it to be continuous and piecewise-differentiable (this way, we can define it piecewisely, like in 3.3.1). As previously said, all the α-relevant arguments must have $S_{arg}(a_i) > \frac{lb+ub}{2}$, so this function could be defined in the interval$[\frac{lb+ub}{2}, ub]$, considering $f(\frac{lb+ub}{2}) = 0$, because conveniently we start at the minimum of the codomain.

Note that all the conditions demanded to these functions are the same as the ones demanded for an importance function (see Definition 3.1.2) only that we only consider the interval $[\frac{lb+ub}{2}, ub]$. Hence:

Definition 4.2.1. *$I_{arg}(x)$ is an **argument importance function** if is an importance function restricted to the interval $[\frac{lb+ub}{2}, ub]$.*

Once the argument importance function I_{arg} is defined, it becomes natural to consider q as:

$$q = \Big(\frac{I_{arg}(S_{arg}(a_{\sigma(1)}))}{T}, \ldots, \frac{I_{arg}(S_{arg}(a_{\sigma(k')}))}{T}\Big)$$

where $T = \sum_{i=1}^{k'} I_{arg}(S_{arg}(a_{\sigma(i)}))$, $a_{\sigma(i)} \in R_\alpha(A) = \{a_1, \ldots, a_{k'}\}$ and $a_{\sigma(i)}$ is the α-relevant argument with the i^{th} largest support. Note that, we have to order the argument's supports because the WOWA orders them when it applies this vector, hence, this way the WOWA will apply each weight to its corresponding argument support.

With all this defined, we can now give the definition for the support function:

Definition 4.2.2. *Let λ be an opinion spectrum, an* ***argument set support function*** *S_{set} is a function that takes a non-empty argument set A, with $R_\alpha(A) \neq \emptyset$, and gives its support in λ as follows:*

$$S_{set}(A) = S_{set}(R_\alpha(A)) = WOWA_{w,q}(S_{arg}(a_1), \ldots, S_{arg}(a_{k'}))$$

with $a_i \in R_\alpha(A) = \{a_1, \ldots, a_{k'}\}$ and where:

$$w = \Big(\frac{\sum_{j=1}^{dim(\vec{O}_{a_1})} I_{opinion}(o_j^1)}{\tau}, \ldots, \frac{\sum_{j=1}^{dim(\vec{O}_{a_{k'}})} I_{opinion}(o_j^{k'})}{\tau}\Big)$$

$$with\ \tau = \sum_{i=1}^{k'} \Big(\sum_{j=1}^{dim(\vec{O}_{a_i})} I_{opinion}(o_j^i)\Big)\ with\ o_j^i \in \vec{O}_{a_i} = \{o_1^i, \ldots o_{n_i}^i\}$$

and

$$q = \Big(\frac{I_{arg}(S_{arg}(a_{\sigma(1)}))}{T}, \ldots, \frac{I_{arg}(S_{arg}(a_{\sigma(k')}))}{T}\Big)$$

where $T = \sum_{i=1}^{k'} I_{arg}(S_{arg}(a_{\sigma(i)}))$, $a_{\sigma(i)} \in R_\alpha(A) = \{a_1, \ldots, a_{k'}\}$ and $a_{\sigma(i)}$ is the α-relevant argument with the i^{th} largest support.

Comment. *If there are no α-relevant arguments then we cannot asses the support for the set, hence we consider $S_{set}(\emptyset)$ to be not defined.*

4.3 Argument importance function construction and alpha choice

In order to choose an argument importance function, we can take the opinion importance function used in the argument support process and restrict it to the part where $x \in \left[\frac{lb+ub}{2}, ub\right]$. For example, to define the natural argument importance function we would have to take the equitable natural opinion importance function restricted to $x \in \left[\frac{lb+ub}{2}, ub\right]$. Figure 4.1 shows an example of the natural argument importance function in the spectrum $\lambda = [1, 5]$, this example has been constructed from the equitable natural opinion importance function in Figure 3.2.

The choice of the α for the α-relevant arguments is complex and depends on the problem and the character given to the whole process. The higher the α, the less arguments will be α-relevant. For the case of a virtual community such as the one described in Chapter 6 we could suppose that $\alpha = 0.3$.

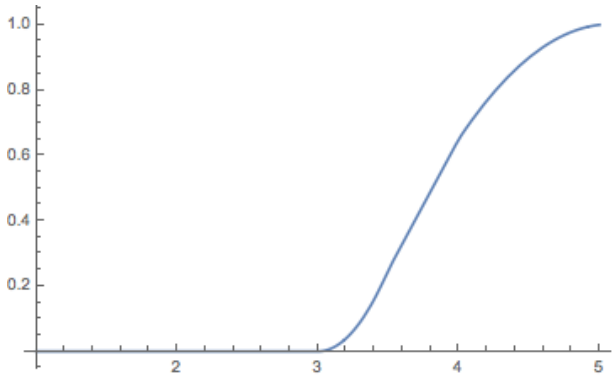

Figure 4.1: Example of a natural argument importance function, in this case $\lambda = [1, 5]$.

Chapter 5

Norm support

Now we face the problem of assessing the support for a norm n. It is worth recalling that we consider a norm n to be associated to two sets, the positive one A_n^+ and the negative one A_n^-. Furthermore, we have proposed a calculation of their support $S_{set}(A_n^+)$ and $S_{set}(A_n^-)$ and we know the opinions of their α-relevant arguments $\vec{O}_{a_i}$ with $a_i \in R_\alpha(A_n^+)$ and $\vec{O}_{\overline{a}_j}$ with $\overline{a}_j \in R_\alpha(A_n^-)$. Therefore, we aim at defining a method to asses the support for n using this information.

5.1 Norm support function

Likewise in previous chapters, we define a support function for the object under discussion, which in this case corresponds to a norm support function. Considering the meaning of all the information we have: Large values in the support for the (relevant) positive arguments of any norm n (that is, $S_{set}(A_n^+)$) should impact positively on its support. On the contrary, large values in the support for the (relevant) negative arguments of any norm n (that is, $S_{set}(A_n^-)$) should impact negatively on its support, meaning that they are detrimental for the enactment of this norm. So the numbers we have to aggregate are $S_{set}(A_n^+)$ and $ub+lb-S_{set}(A_n^-)$, the symmetric value of the support in the spectrum with respect to the center of the spectrum. Figure 5.1 shows the two values that have to be aggregated and the difference between these values and the argument set supports with a graphical example.

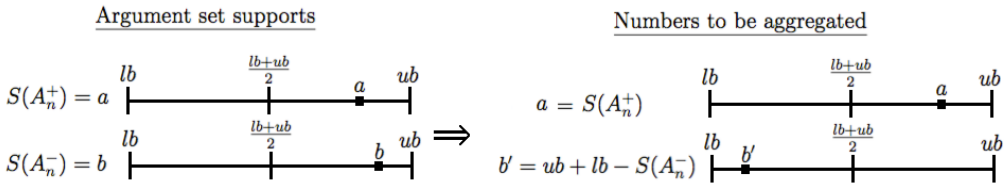

Figure 5.1: Example of the aggregation numbers to asses the norm support and the difference between these and the argument set supports.

We will separate the task of defining the norm support function into different cases:

5.1.1 Case $R_\alpha(A_n^+) \neq \emptyset$ and $R_\alpha(A_n^-) \neq \emptyset$

In case $R_\alpha(A_n^+) \neq \emptyset$ and $R_\alpha(A_n^-) \neq \emptyset$, then $S_{set}(A_n^+)$ and $S_{set}(A_n^-)$ are defined, so, since they have numerical value, we can aggregate them.

We have to aggregate the two values previously explained, in order to do so, we have to consider the overall importance of the sources because the importance of the opinions received by the arguments in each argument set might be different (that is, one argument set can have more important arguments than the other). Also, as in the case of assessing the support for an argument set, we have to weigh the importance of the aggregation values themselves, for the same reason as in the previous chapter(i.e, extreme values are more important because represent stronger (in favor/against) support than middle values considered to be neutral). So the aggregation operator required is a WOWA.

First, to define the q vector of the WOWA(the one weighting the aggregation values), we have to define an argument set importance function to weigh the importance of the value of the argument sets' support. Note that, the values that have to be aggregated lay in $[lb, ub]$. Hence, the argument set importance function is, in fact, an importance function as defined in Definition 3.1.2:

Definition 5.1.1. *Let A be an argument set, $I_{set}(x)$ is an **argument set importance function** if it is an importance function.*

So the q vector of the WOWA should be:

$$q = \Big(\frac{I_{set}(S_{set}(A_n^+))}{T}, \frac{I_{set}(ub + lb - S_{set}(A_n^-))}{T}\Big)$$

$$\text{where } T = I_{set}(S_{set}(A_n^+)) + I_{set}(ub + lb - S_{set}(A_n^-))$$

Comment. *Note that, since we always have $S_{set}(A_n^+) > ub + lb - S_{set}(A_n^-)$, we do not have to order the q vector as in the case of the q vector in the WOWA of the argument set support.*

Regarding the weighting of the overall importance of the two sets, in the previous chapter we introduced α-relevant arguments, and the supports $S_{set}(A_n^+)$ and $S_{set}(A_n^-)$ are calculated based on the supports for these α-relevant arguments, so the proposed w vector of the WOWA has as weights the sum of the importance of the α-relevant arguments' opinions of each set, then the vector has to be divided by the appropriate number, so its elements altogether sum one(required to be a weighting vector). Hence, if we have $R_\alpha(A_n^+) = \{a_1, \ldots, a_{k_1}\}$ and $R_\alpha(A_n^-) = \{\overline{a}_1, \ldots, \overline{a}_{k_2}\}$ then:

$$w = \Big(\frac{\sum_{i=1}^{k_1}(\sum_{j=1}^{n_i} I_{opinion}(o_j^i))}{\tau}, \frac{\sum_{i=1}^{k_2}(\sum_{j=1}^{\overline{n_i}} I_{opinion}(\overline{o}_j^i))}{\tau}\Big)$$

where $\tau = \sum_{i=1}^{k_1}(\sum_{j=1}^{n_i} I_{opinion}(o_j^i)) + \sum_{i=1}^{k_2}(\sum_{j=1}^{\overline{n_i}} I_{opinion}(\overline{o}_j^i))$, o_j^i is the j^{th} opinion in $\vec{O}_{a_i} = \{o_1^i, \ldots, o_{n_i}^i\}$, $a_i \in R_\alpha(A_n^+)$ and $\overline{o}_j^i$ is the j^{th} opinion in $\vec{O}_{\overline{a}_i} = \{\overline{o}_1^i, \ldots, \overline{o}_{\overline{n_i}}^i\}$, $\overline{a}_i \in R_\alpha(A_n^-)$.

With the WOWA vectors defined we can proceed with the definition of the norm support function:

Definition 5.1.2. *A **norm support function** is a function $S_{norm}(n)$ that takes a norm n (in this case, suppose $R_\alpha(A_n^+) \neq \emptyset$ and $R_\alpha(A_n^-) \neq \emptyset$) and using its argument set's supports assesses the support for the norm in $\lambda = [lb, ub]$ as follows:*

$$S_{norm}(n) = WOWA_{w,q}(S_{set}(A_n^+), ub + lb - S_{set}(A_n^-))$$

Being w:

$$w = \Big(\frac{\sum_{i=1}^{k_1}(\sum_{j=1}^{n_i} I_{opinion}(o_j^i))}{\tau}, \frac{\sum_{i=1}^{k_2}(\sum_{j=1}^{\overline{n_i}} I_{opinion}(\overline{o}_j^i))}{\tau}\Big)$$

where $\tau = \sum_{i=1}^{k_1}(\sum_{j=1}^{n_i} I_{opinion}(o_j^i)) + \sum_{i=1}^{k_2}(\sum_{j=1}^{\overline{n_i}} I_{opinion}(\overline{o}_j^i))$, o_j^i is the j^{th} opinion in $\vec{O}_{a_i} = \{o_1^i, \ldots, o_{n_i}^i\}$, $a_i \in R_\alpha(A_n^+) = \{a_1, \ldots, a_{k_1}\}$ and $\overline{o}_j^i$ is the j^{th} opinion in $\vec{O}_{\overline{a}_i} = \{\overline{o}_1^i, \ldots, \overline{o}_{\overline{n_i}}^i\}$, $\overline{a}_i \in R_\alpha(A_n^-) = \{\overline{a}_1, \ldots, \overline{a}_{k_2}\}$.

And being q:

$$q = \Big(\frac{I_{set}(S_{set}(A_n^+))}{T}, \frac{I_{set}(ub + lb - S_{set}(A_n^-))}{T}\Big)$$
$$where\ T = I_{set}(S_{set}(A_n^+)) + I_{set}(ub + lb - S_{set}(A_n^-))$$

Now that we have defined a way to calculate $S_{norm}(n)$, the norm n has a support in λ, so we can decide if n has to be enacted or not. In general, n will be applied if $S_{norm}(n) \geq \mu$, where $\mu \in (\frac{lb+ub}{2}, ub]$ is the minimum approval support threshold. This μ may be different depending on the conditions related to each problem. Note that it is a necessary condition for μ to be in $(\frac{lb+ub}{2}, ub]$, because if we want to apply a norm at least it has to have a support laying on the pro block of the spectrum.

5.1.2 Other cases

$\underline{R_\alpha(A_n^+) = \emptyset \textbf{ and } R_\alpha(A_n^-) = \emptyset}$

When both $R_\alpha(A_n^+)$ and $R_\alpha(A_n^-)$ are $\emptyset$, $S_{norm}(n)$ cannot be calculated with the WOWA, as in the previous section, because the argument set supports are not a number, hence, in this case, we define $S_{norm}(n)$ to be not defined, so the norm is not enacted.

$\underline{R_\alpha(A_n^+) = \emptyset \textbf{ and } R_\alpha(A_n^-) \neq \emptyset}$

If only $R_\alpha(A_n^+) = \emptyset$, then $S_{set}(A_n^+)$ is not defined, hence we define $S_{norm}(n) = ub+lb-S_{set}(A_n^-)$, since the only numerical value we can aggregate is $ub+lb-S_{set}(A_n^-)$. In this case, since $ub + lb - S_{set}(A_n)^- \leq \frac{ub+lb}{2}$, the norm is considered not having positive support and therefore it is not applied.

$\underline{R_\alpha(A_n^+) \neq \emptyset \textbf{ and } R_\alpha(A_n^-) = \emptyset}$

Finally, in case only $R_\alpha(A_n^-) = \emptyset$, then $S_{set}(A_n^-)$ is not defined, so $ub+lb-S_{set}(A_n^-)$ is not defined either, hence we define $S_{norm}(n) = S_{set}(A_n^+)$, since the only numerical value we can aggregate is $S_{set}(A_n^+)$. In this case, since $S_{set}(A_n^+) > \frac{ub+lb}{2}$, the norm will be applied if $S_{norm}(n) = S_{set}(A_n^+) \geq \mu$, where $\mu \in (\frac{ub+lb}{2}, ub]$ is the minimum approval support threshold.

5.2 Detailed example of the overall norm support assessment process

In Chapter 6 we will compare the proposed norm support assessment method with other possible approaches, but before comparing it, we will apply it to an example case. Thus, the proposed method can be understood clearly.

Comment. *Since in this example there will be vectors with real numbers, I will use a point for denoting a number's decimals (n.dddd), by doing this, possible confusions with the commas of the vectors are avoided.*

Example. *Suppose a norm n defined in the spectrum $\lambda = [1, 5]$, with the following arguments:*

- *Positive arguments:*
  - *posarg1 with opinions: 5, 5, 5, 4.8, 5, 4.8, 4*
  - *posarg2 with opinions: 4, 3.5, 3.5, 4.5, 4*
  - *posarg3 with opinions: 1, 1, 2, 1, 2, 3, 1, 1*
- *Negative arguments:*
  - *negarg1 with opinions: 4, 3.5, 5, 4, 3.5, 4, 4, 4, 4*
  - *negarg2 with opinions: 4, 4*

Comment. *For the aggregation, we will use the equitable version of the natural opinion importance function, defined in Section 3.3.1, as both the opinion importance function and the argument set importance function, and the same function restricted to $[\frac{lb+ub}{2}, ub]$ as the argument importance function. As for the α in α-relevant arguments, we will use $\alpha = 0.3$.*

First, we have to assess the support for each argument:

- posarg1:
  We have the opinion importances: $I_{opinion}(5) = 1$, $I_{opinion}(4.8) = 0.978$, $I_{opinion}(4) = 0.65$. So, i this case, the vector of the weighted mean is:
  $w = (\frac{1}{6.06}, \frac{1}{6.06}, \frac{1}{6.06}, \frac{0.978}{6.06}, \frac{1}{6.06}, \frac{0.978}{6.06}, \frac{0.65}{6.06})$. Hence the support for the argument is: $S_{arg}(posarg1) = WM_w(5, 5, 5, 4.8, 5, 4.8, 4) = 4.8424$

- posarg2:
  We have the opinion importances: $I_{opinion}(4) = 0.65$, $I_{opinion}(3.5) = 0.25$, $I_{opinion}(4.5) = 0.9$. So, in this case, the vector of the weighted mean is:
  $w = (\frac{0.65}{2.7}, \frac{0.25}{2.7}, \frac{0.25}{2.7}, \frac{0.9}{2.7}, \frac{0.65}{2.7})$. Hence the support for the argument is: $S_{arg}(posarg2) = WM_w(4, 3.5, 3.5, 3.5, 4) = 4.0741$

- posarg3:
  We have the opinion importances: $I_{opinion}(1) = 1$, $I_{opinion}(2) = 0.65$, $I_{opinion}(3) = 0$. So, in this case, the vector of the weighted mean is:
  $w = (\frac{1}{6.3}, \frac{1}{6.3}, \frac{0.65}{6.3}, \frac{1}{6.3}, \frac{0.65}{6.3}, 0, \frac{1}{6.3}, \frac{1}{6.3})$. Hence the support for the argument is: $S_{arg}(posarg3) = WM_w(1, 1, 2, 1, 2, 3, 1, 1) = 1.2063$

- negarg1:
  We have the opinion importances: $I_{opinion}(4) = 0.65$, $I_{opinion}(3.5) = 0.25$, $I_{opinion}(5) = 1$. So, in this case, the vector of the weighted mean is:
  $w = (\frac{0.65}{5.4}, \frac{0.25}{5.4}, \frac{1}{5.4}, \frac{0.65}{5.4}, \frac{0.25}{5.4}, \frac{0.65}{5.4}, \frac{0.65}{5.4}, \frac{0.65}{5.4}, \frac{0.65}{5.4})$. Hence the support for the argument is: $S_{arg}(negarg1) = WM_w(4, 3.5, 5, 4, 3.5, 4, 4, 4, 4) = 4.1389$

- negarg2:
  We have the opinion importances: $I_{opinion}(4) = 0.65$. So, in this case, the vector of the weighted mean is:
  $w = (\frac{0.65}{1.3}, \frac{0.65}{1.3})$. Hence the support for the argument is: $S_{arg}(negarg2) = WM_w(4, 4) = 4$

Now, the next step is to calculate $S_{set}(A_n^+)$ and $S_{set}(A_n^-)$, to do so, first we have to determine $R_\alpha(A_n^+)$ and $R_\alpha(A_n^-)$. In this case, $\alpha = 0.3$, and since the argument with most opinions is negarg1 with 9 opinions, for an argument to be α-relevant it has to have at least $9 \cdot 0.3 = 2.7$ opinions (3 opinions) and has to have a score greater than 3, so the α-relevant argument sets are: $R_{0.3}(A_n^+) = \{posarg1, posarg2\}$ and $R_{0.3}(A_n^-) = \{negarg1\}$. Note that, since negative arguments have only one α-relevant argument, the support for the negative argument set is $S_{set}(A_n^-) = 4.1389$, because is an aggregation of only one element. For the support for the positive arguments, we have to aggregate the supports for the two α-relevant arguments (that is, $S_{arg}(posarg1) = 4.8424$ and $S_{arg}(posarg2) = 4.0741$) with the WOWA. Firstly we have to calculate the w vector using the importance of the opinions in each argument:

$$w = \Big(\frac{4I_{opinion}(5) + 2I_{opinion}(4.8) + I_{opinion}(4)}{\tau}, \frac{2I_{opinion}(3.5) + 2I_{opinion}(4) + I_{opinion}(4.5)}{\tau}\Big)$$

$$= \Big(\frac{4 + 2 \cdot 0.978 + 0.65}{4 + 2 \cdot 0.978 + 3 \cdot 0.65 + 2 \cdot 0.25 + 0.9}, \frac{2 \cdot 0.25 + 2 \cdot 0.65 + 0.9}{4 + 2 \cdot 0.978 + 3 \cdot 0.65 + 2 \cdot 0.25 + 0.9}\Big)$$

$$= \Big(\frac{6.606}{9.306}, \frac{2.7}{9.306}\Big) = \Big(\frac{367}{517}, \frac{150}{517}\Big)$$

Secondly, we have to calculate the q vector using the argument importance function and the ordered supports for the arguments(that is, $S_{arg}(a_{\sigma(1)}) = S_{arg}(posarg1) = 4.8424$ and $S_{arg}(a_{\sigma(2)}) = S_{arg}(posarg2) = 4.0741$, hence:

$$q = \Big(\frac{I_{arg}(4.8424)}{I_{arg}(4.8424) + I_{arg}(4.0741)}, \frac{I_{arg}(4.0741)}{I_{arg}(4.8424) + I_{arg}(4.0741)}\Big)$$

$$= \Big(\frac{0.984669}{0.984669 + 0.696518}, \frac{0.696518}{0.984669 + 0.696518}\Big) \approx (0.5857, 0.4143)$$

With the two vectors defined the positive argument set support can be assessed as (the WOWA is calculated using [6]):

$$S_{set}(A_n^+) = WOWA_{w,q}(4.8424, 4.0741) = 4.6701$$

Since we have the supports for the two argument sets, we can now assess the support for the norm. To calculate the support for the norm, the two values to aggregate are: $S_{set}(A_n^+) = 4.6701$ and $ub + lb - S_{set}(A_n^-) = 5 + 1 - 4.1389 = 1.8611$. Since we have to do a WOWA, first we have to define the two weighting vectors for this aggregation. In the case of q:

$$q' = \Big(\frac{I_{set}(4.6701)}{I_{set}(4.6701) + I_{set}(1.8611)}, \frac{I_{set}(1.8611)}{I_{set}(4.6701) + I_{set}(1.8611)}\Big) =$$

$$\Big(\frac{0.950855}{0.950855 + 0.734497}, \frac{0.734497}{0.950855 + 0.734497}\Big) \approx (0.56419, 0.43581)$$

And w:

$$w' = \Big(\frac{4 + 2 \cdot 0.978 + 3 \cdot 0.65 + 2 \cdot 0.25 + 0.9}{T}, \frac{6 \cdot 0.65 + 2 \cdot 0.25 + 1}{T}\Big)$$

$$= \Big(\frac{9.306}{9.306 + 5.4}, \frac{5.4}{9.306 + 5.4}\Big) = \Big(\frac{517}{817}, \frac{300}{817}\Big)$$

With the two weighting vectors for the WOWA, the norm support can be assessed as follows (the WOWA is calculated using [6]):

$$S_{norm}(n) = WOWA_{w',q'}(4.6701, 1.8611) = 3.8010$$

Chapter 6

Particular Case: The social network

In this chapter we will apply the norm argument map to a social network. Additionally, for this particular case study, both the aggregation process and support assessment, will be explained and compared to other methods such as a naive average method.

6.1 Formalizing the problem

In [5] the author explains the way the social network is implemented, participants can upload posts, complain about other participants' posts and, in case a post is detected as being conflictive, can create a norm to regulate that conflict. As a result, a discussion is initiated and participants can argue about the norm by using a norm argument map. Finally, when the discussion ends, the norm support is evaluated and it is decided weather the norm is enacted (i.e. if it becomes active) or not in the social network. Figure 6.1 shows an example of the norm argument map with a norm under discussion as described in [5].

To formalize this problem, we will define the components of the norm argument map defined in this project. We will consider that participants can give opinions to the arguments within the real interval $\lambda = [1, 5]$[1] this spectrum could represent a 5-star rating system which might be the more commonly used way to give ratings, since Amazon, iTunes, Netflix, TripAdvisor, Yelp and other webs use it. Note that, the meaning of the opinions in this interval is the one reflected in Figure 3.1. In the case of this implementation of the norm argument map the classification function κ is in fact computed by the participants themselves who, when adding a new

[1]Note that, in [5] there is a difference between the opinions participants can assign to an argument (in that case, $\{1, 2, 3, 4, 5\}$) and the spectrum in which supports are defined (in that case, $[1, 5]$). This was done to simplify the interface, which is implemented with a 5-star rating system where participants can click over the stars to assign them to the argument associated.

argument, choose the appropriate destination set (in favor/against the norm). As for the aggregation, like in the example of Section 5.2, we will use the equitable version of the natural opinion importance function, defined in Section 3.3.1, as both the opinion importance function and the argument set importance function, and the same function restricted to $[\frac{lb+ub}{2}, ub]$ as the argument importance function.

6.2 Comparing aggregation methods

In this section we will do some comparisons to ensure that the aggregation process proposed in this project gives better results than other approaches, note that all the cases will be in the spectrum $\lambda = [1, 5]$, as previously stated. Specifically, two aggregation methods are compared: the one proposed in this project[2] and a naive average approach. We will also discuss why the method in [5] is not suitable either.

6.2.1 Comparison with the naive average approach

The norm support of this naive approach is calculated by averaging all the opinions of all the arguments. Opinions from arguments against the norm ($\overline{o}^i_j$) will be aggregated as $ub + lb - \overline{o}^i_j$, the symmetric value in the spectrum, as depicted in previous Figure 5.1 in Section 5.1.

Example. *The norm $n1$ has an argument in favor $posarg1$ (with an opinion of 5) and an argument against $negarg1$ (with an opinion of 5), if we apply the naive average approach the norm has a resulting support of $\frac{5+1}{2} = 3$, since an opinion value of 5 about a negative argument is mapped into 1, its symmetric value in the spectrum.*

Notation. *Let n be a norm. In this section we will use $S_{norm}(n)$ as the norm support function defined in the project and $S_{avg}(n)$ as the norm support calculated with the naive average approach.*

Being naive approach defined, we can proceed with the comparisons. First, we will show some cases in which the two methods give the same result and ratify its validity:

Comparison 1. *In the case of the previous example, and in extent, any case with two arguments (one positive and one negative) with an equal number of opinions having value 5, we have $S_{avg}(n) = S_{norm}(n) = 3$. This happens because this case represents an split opinion state, and since none of the extremes has advantage over the other the resulting support is neutral.*

[2]Calculations have been done and can be replicated with the code attached to the project or referenced in the appendices.

Comparison 2. *In the case where there are only positive arguments with maximum opinion values, then* $S_{avg}(n) = S_{norm}(n) = 5$. *This case represents the unanimity case, if everyone thinks the same the support for the norm has to be what everyone thinks. Note that, analogously, if there are only negative arguments with maximum opinion values, then* $S_{avg}(n) = S_{norm}(n) = 1$ *for the same reason.*

Now, we will show the cases in which the two methods give different results and discuss about the reasons for our proposed method to provide a more accurate or meaningful result:

Comparison 3. *Consider the next arguments and opinions for norm n:*

- *posarg1 with opinions: 3.5, 3.25, 3.5, 3, 2.5*
- *negarg1 with opinions: 1, 1, 1.2, 1.3, 1.25*

This case represents a weak positive argument, in the sense that people are neutral about it, and a bad negative argument. In this case $S_{avg}(n) = 4$, *whereas* $S_{norm}(n) = 3,1731$, *note that if we required a minimum support of 4 for applying the norm and we used the average naive approach, we would apply the norm, although it does not have a single positive argument with a single positive opinion greater than 4. This happens because the average approach does not use* α*-relevant arguments. In the case of the project's method the support is more similar to the one expressed in the arguments: a neutral support. Analogously, if considering this same example but with the opinions interchanged,* $S_{avg}(n) = 2$ *and* $S_{norm}(n) = 2,8269$, *so the project's method is still neutral whereas now the average approach gives a low support.*

A big difference in support happens when arguments are bad and are not enough, such as in this next cases:

Comparison 4. *Consider the next arguments and opinions for norm n:*

- *posarg1 with opinions: 2, 2.5, 1, 3, 2.8*
- *posarg2 with opinions: 1, 1, 1.2, 1.3, 1.25*
- *no negative arguments*

In this case $S_{avg}(n) = 1,705$, *but the fact that there are no good arguments does not imply that the norm is not good, because for example in this case, if the norm was not a good norm maybe someone would have given a good negative argument, so maybe the debating period should last more, that way people could give more potentially strong arguments, that is why the support* $S_{norm}(n)$ *not defined is more adequate.*

Analogously:

Comparison 5. *Consider the next arguments and opinions for norm n:*

- *no positive arguments*
- *negarg1 with opinions: 1, 1, 1.2, 1.3, 1.25*

In this case $S_{avg}(n) = 4,85$, but the fact that there is a bad negative argument does not mean that the norm is validated, in fact, as in the previous comparison, this means that maybe the debate should go on, hence in this case $S_{norm}(n)$ not defined is also an adequate support.

A combination of the two previous comparisons (with both bad positive and negative arguments) would be the same case, $S_{avg}(n)$ would give a not adequate support whereas, $S_{norm}(n)$ would be not defined, which means that maybe the debate should go on.

Another case where the two methods give fairly different supports:

Comparison 6. *Let n be a norm with the following arguments and opinions:*

- *posarg1 with opinions: 5*
- *posarg2 with opinions: 5, 5*
- *negarg1 with opinions: 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1*
- *negarg2 with opinions: 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1*

In this case $S_{avg}(n) = 5$, but note that as previously stated a bad negative argument should not support favorably the norm. Also, the positive arguments have not received big amounts of opinions. If the positive arguments were good enough they would have received more opinions, hence in this case the norm should have a not defined support, which is the case of $S_{norm}(n)$.

Comparison 7. *Let the arguments of norm n be:*

- *posarg1 with opinions: 5, 5, 5*
- *negarg1 with opinions: 3.15, 3.2, 2.8*
- *negarg2 with opinions: 3, 3.5, 2.6*
- *negarg3 with opinions: 2.5, 3.5, 3.2*

In this case $S_{avg}(n) = 3,5375$, whereas $S_{norm}(n) = 4,9842$. Note that, in this case there is a strong positive argument and a few weak or neutral negative arguments, so since people have not found an argument sufficiently strong to attack the norm but they have found a strong argument to enact it, the support should be favorable to the enacting of the norm. In this case if the minimum support to apply a norm was 4, the average method would revoke the norm whereas the proposed method would accept it. This happens because it is fundamental to weigh the importance of the arguments as well as the importance of the argument sets, this way neutral arguments do not weigh much in the overall norm support.

6.2.2 Discussion about the method in [5]

Suppose a discussion of the norm n with the norm argument map in the spectrum $[1, 5]$. In summary, the method to asses the norm support proposed in [5] consists of the following steps (for the sake of clarity, the notation introduced along this work is used):

- From the two argument sets of the norm argument map ($A_n^+ = \{a_1, \ldots, a_p\}$ and $A_n^- = \{\overline{a_1}, \ldots, \overline{a_q}\}$), the vectors μ^n and $\overline{\mu}^n$ are constructed as follows:

$$\mu^n = (m_1^n, \ldots, m_p^n), \text{ with } m_i^n = median(\vec{O}_{a_i}) \ \forall a_i \in A_n^+$$

$$\overline{\mu}^n = (\overline{m}_1^n, \ldots, \overline{m}_q^n), \text{ with } \overline{m}_i^n = median(\vec{O}_{\overline{a}_i}) \ \forall \overline{a}_i \in A_n^-$$

- Then, using two weighting vectors ω and $\overline{\omega}$ (the construction of which is detailed in [5]) two OWA's are calculated: $OWA_\omega(\mu^n)$ and $OWA_{\overline{\omega}}(\overline{\mu}^n)$
- Finally the norm support is evaluated as: $OWA_\omega(\mu^n) - OWA_{\overline{\omega}}(\overline{\mu}^n)$

Now that we have a general idea of the method, we can start discussing. Firstly, although this method does not consider the terms "argument support" or "argument set support" but, from the definition of the method, it is clear that the support for an argument consists of the median of its opinions, this is not a suitable way of assessing the support to an argument since, for example, if an argument a_i had $\vec{O}_{a_i} = (5, 5, 5, 1, 1, 1, 1)$, then its support would be 1, although in reality the argument has a more neutral support.

From the definition we can also establish that the argument set support is defined as an OWA, this is not suitable either because the importance of the arguments(that is, the sum of the importance of its opinions) in the argument sets are not weighted, only the values of the arguments' supports are weighted. This is problematic because, for example, an argument with support 5 and a single opinion can weigh more than another argument with support 4 and 1000 value 4 opinions.

Finally, the norm support is evaluated as $OWA_\omega(\mu^n) - OWA_{\overline{\omega}}(\overline{\mu}^n)$. This happens to be problematic too, because, for example, if $OWA_\omega(\mu^n) = 1$ and $OWA_{\overline{\omega}}(\overline{\mu}^n) = 5$, then the support for the norm would be -4 which is out of the spectrum. This is depicted in Figure 6.1, which shows an example of a norm being discussed, note that, in that example, the norm has a support of $\frac{1}{2}$ stars, which is out of the spectrum, since the minimum value possible is one star. Furthermore, if $OWA_\omega(\mu^n) = 5$ and the positive arguments had lots of important opinions and $OWA_{\overline{\omega}}(\overline{\mu}^n) = 4$ and the negative arguments had little opinions (hence, little important opinions), then the support for the norm would be 1 although it should be higher, since positive arguments have maximum support and bigger number of important opinions than the negative ones.

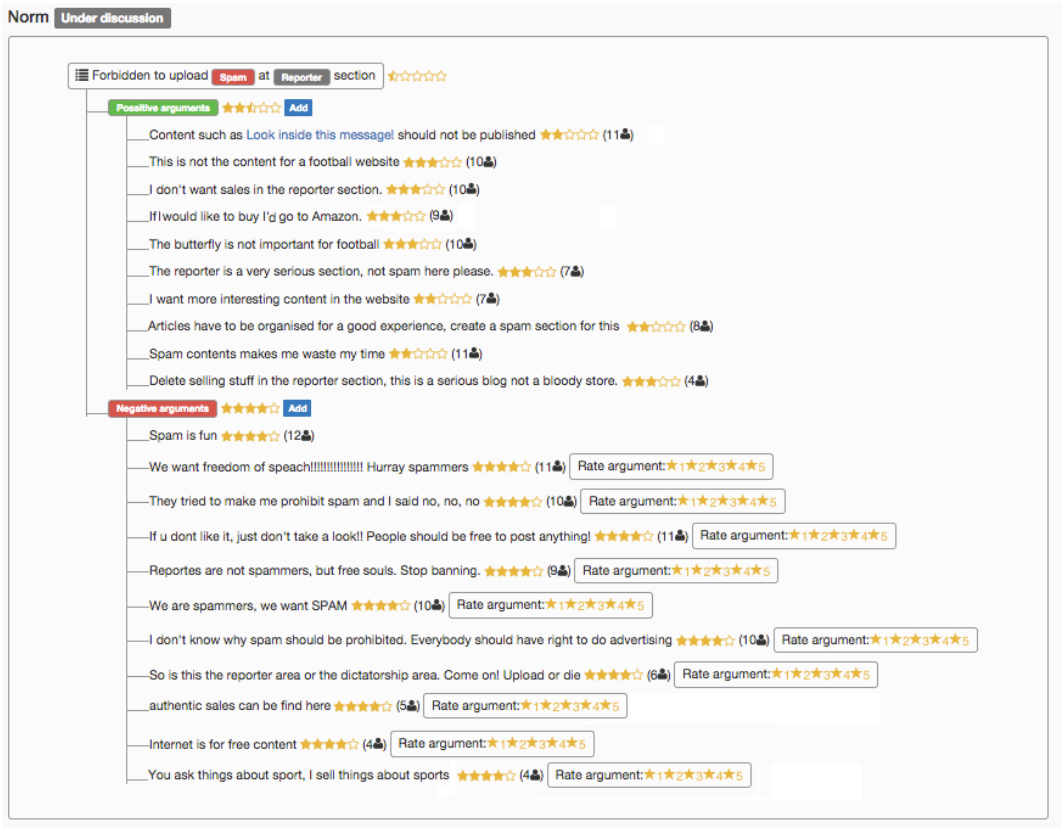

Figure 6.1: Example of a norm discussion in [5].

Chapter 7

Conclusions and future work

In this project we have formalized and studied several objects achieving the expected goals successfully. To sum up, the conclusions of this work are the following:

7.1 Conclusions

In order to provide a more democratic way of moderating virtual communities, we have formalized the idea of the norm argument map by providing all the necessary definitions. From that definitions, we divided the problem of assessing the support for a norm from the community participants' opinions, into three subsequent subproblems related to subsequent elements. Next, by finding the way of computing the support for each element in the norm argument map, we finally found the way of giving the support on the debated norm.

These three subproblems also resulted in new definitions and objects in the overall process. Thus, the task of calculating the support for a single argument has resulted in the creation of the opinion importance function, so we can give different importances to the opinions based on their meaning. Also, the research of a method for aggregating argument's support into a support for an argument set has lead us to the definition of relevant and α-relevant arguments, hence reducing the number of arguments to aggregate and discarding the ones that with little debate could distort the overall support.

Finally, we compared the work done in this project to both a naive and previous approaches on the subject and found that both of them, although can give similar results in some cases, on others can also be very unreliable.

Regarding other matters of concern, it may be worth mentioning that the semantics given to the opinions have been established in a way that arguments could be accepted or criticized, this way all points of view can be reflected into an opinion value.

A particular case of this project would be to consider that the minimum of the opinion spectrum represents a neutral opinion and the nearer to the maximum the more accepted the argument is. In this case all the work done could be reused since, for example, the importance function in this new semantic could be as in Definition 3.1.2, but keeping conditions $(C1)$ and $(C2)$ and changing condition $(C3)$ for $(C3')I(lb) = 0$ and condition $(C4)$ for $(C4')I'(x) > 0$ if $x \in \lambda$ and I differentiable in x.

7.2 Future work

The following are some points that could be done or improved:

- As said previously, in this project, we have considered all arguments to be different, this assumption can lead to a debating process with several similar or equal arguments that individually could not impact the support for the norm debated, but altogether impact crucially to the norm's support. To solve this problem, we could rely on moderators to collapse similar arguments, but since the aim of the norm argument map is to implement a system without moderators, the only way of solving this problem would be to incorporate a Natural Language Processing (NLP) system which could recognize similar arguments and collapse them automatically.

- In this project we have applied the method to decide norms for virtual communities, but norm argument maps can also be applied to other real world cases, such as citizen's juries, direct democracy debates, or referendums.

- Due to a lack of time and resources no tests involving humans have been performed for the proposed method. Nevertheless, tests involving humans should be done in order to truly assess the reliability of the approach.

- Further aspects of the deliberation process should be studied, for example, the decision about when to stop the debating process or what to do when a debating process ends with the norm having a not defined support.

Bibliography

[1] Edmond Awad, Richard Booth, Fernando Tohmé, and Iyad Rahwan. Judgement aggregation in multi-agent argumentation. *Journal of Logic and Computation*, aug 2015.

[2] M Detyniecki. Fundamentals on aggregation operators. *This manuscript is based on Detyniecki's doctoral ...*, 2001.

[3] Mark Klein. Enabling Large-Scale Deliberation Using Attention-Mediation Metrics. *Computer Supported Cooperative Work (CSCW)*, 21(4-5):449–473, 2012.

[4] Miguel Sánchez-Mazas. Cálculo de las normas. pages 125–180, 1973.

[5] David Sánchez Pinsach. Norms crowdsourcing. *Master thesis, http://upcommons.upc.edu/handle/2099.1/26012*, may 2015.

[6] V Torra. Java implementation for WOWA, OWA and WM, http://www.mdai.cat/ifao/wowa.php.

[7] V Torra. The weighted OWA operator. *International Journal of Intelligent Systems*, 12(2):153–166, feb 1997.

[8] V Torra. The WOWA operator and the interpolation function W*: Chen and Otto's interpolation method revisited. *Fuzzy Sets and Systems*, 2000.

[9] V Torra. The WOWA Operator: A Review. In *Recent Developments in the Ordered Weighted Averaging Operators: Theory and Practice*, pages 17–28. Springer, 2011.

[10] V Torra and Z Lv. On the WOWA operator and its interpolation function. *International Journal of Intelligent Systems*, 2009.

[11] V. Torra and Y. Narukawa. *Modeling decisions: information fusion and aggregation operators*. Springer, 2007.

[12] RR Yager. On ordered weighted averaging aggregation operators in multicriteria decisionmaking. *Systems, Man and Cybernetics, IEEE Transactions ...*, 1988.

Appendices

Appendix A

Norm argument map code and mathematica notebooks

Since one of the goals of this project has been to compare the naive method to the one implemented in this project, I have coded a project in Java (NormSupportEvaluation) with all the needed tools for implementing the method and using it. This code uses the package ww coded by V.Torra that can be found at [6]. Specifically, I have coded a package to implement the norm argument map along with a front end to input data from the command line or from a file. This code can be found attached to the project or uploaded at:

`https://bitbucket.org/msamsa/norm-argument-map.git`

Also the mathematica notebooks of the importance functions explained in Chapter 3 can be found attached to this project or at:

`https://bitbucket.org/msamsa/importance-functions.git`

Appendix B

Norm argument map code user guide

B.1 The namargumentationsystem package

The package namargumentationsystem has all the necessary classes to implement the norm argument map and the aggregation method explained in the associated final research project. In the NormSupportEvaluation folder you can find the doc folder with the javadoc for this package, to view it simply open index.html. Figure B.1 represents the class diagram for this package. Additionally, in the following pages each class and its methods are explained in further detail:

Norm class

The norm is implemented as an object with a string containing the norm's statement. Implementing a fully functional class norm is out of the scope of this project. The object has get and set methods.

Opinion class

The opinion is implemented as an object with a double containing the opinion value. The object has get and set methods.

Spectrum class

The spectrum is implemented as an object with two double variables, one containing the lower bound and the other the upper bound. The constructor method ensures that the lower bound is lower than the upper one, if that is not the case they are interchanged.

Methods

- `public Boolean equals(Spectrum spec)`: gets another Spectrum ob-

ject and returns true if the two spectrums are the same or false otherwise.

- `public double middle()`: returns a double containing the middle point of the spectrum $\frac{lb+ub}{2}$.
- `public Boolean has(double num)`: gets a double number and returns true if the number is in the spectrum or false otherwise.
- `public Boolean has(Opinion o)`: gets an opinion and returns true if the opinion's value is in the spectrum or false otherwise.
- `public double symmetric(double x)`: gets a real number x and returns the number symmetric to the one got in respect to the spectrum's middle $(ub + lb - x)$, if x is not in the spectrum returns NaN.
- `public String toString()`: returns a string containing the representation of the spectrum in the format "[lower bound, upper bound]".
- The class also has get and set methods for the variables.

Importance function class

The importance function class is an abstract class, so that way, a user using the package can either create his own importance function (extending the abstract class) or use the DefaultImportanceFunction class where the natural opinion importance function is implemented.

Methods

- `public abstract double compute(double x)`: receives a double and returns the value of the importance function in that point.

Argument class

An Argument is an object with a string, the argument's statement; a list of Opinion objects, containing the opinions of the argument; a Spectrum, in which the Opinion values must be and an ImportanceFunction object, necessary for assessing the support for the argument. The argument class can be initialized with all of these objects or without an Opinion list.

Methods

- `public void addOpinion(double num)`: Adds a new opinion (using the double num) in the argument's list of opinions. If num is not in the argument's spectrum the method prints an error message and nothing is added.
- `public void addOpinion(Opinion o)`: This method does the same as the previous one but instead of using a double it uses an opinion object.

- `public double support()`: Returns a double containing the support for the argument. The support is computed with the method described in the project and using the argument's importance function. First the vectors of data and weights are created and filled. Then the method computes the support using the weighted mean in the ww package.

- `public Boolean in(Spectrum spec)`: Returns true if spec is the argument's spectrum, false otherwise.

- `public double numberOpinions()`: Returns a double containing the number of opinions received by the argument.

- `public double weight()`: Returns the weight of the argument as the sum of the importance of all its opinions.

- `public void printStatus(PrintStream out)`: Prints the status of the argument as a line of the form "statement support Nbr. of opinions". The method prints it using the PrintStream out.

- The class also has get and set methods for some variables.

ArgumentSet class

The ArgumentSet class has a list of argument objects; a spectrum object (the spectrum of the arguments in it) and an ImportanceFunction object to use it in the process of assessing the support. To create a new ArgumentSet it is only necessary to give the spectrum and the importance function, although the list of arguments can also be given.

Methods

- `public void addArgument(Argument a)`: Adds the argument a into the argument set's argument list, but only if they have the same spectrum.

- `public void addArgument(String s, List<Opinion> o)`: Creates a new Argument object using s and o. Then calls the previous method with the argument created.

- `public ArgumentSet alphaRelevantArguments(double num)`: Returns an argument set with an argument list containing only the α-relevant arguments (as defined in the project), to be considered an α-relevant argument, an argument has to have a support greater than the middle of the spectrum and has to have a number of opinions greater than num(which has to be the product of α and the maximum number of opinions a single argument of the norm has received).

- `public double numberOpinions()`: Returns the number of opinions of all the arguments in the argument set.

- `public double weight(double minimum)`: Returns the weight of the argument set as the sum of the importance of the α-relevant arguments in the set (`this.alphaRelevantArguments(minimum)`).

- `public double weight()`: Returns the weight of the argument set as the sum of the importance of all its arguments.

- `public double numberRelevantOpinions(double numrel)`: Returns the number of opinions of all the arguments in the α-relevant argument set returned by the method `this.alphaRelevantArguments(numrel)`.

- `public double support(double num)`: Returns a double containing the support for the argument set. First the method extracts the α-relevant arguments, then fills the data vector and the two weighting vectors of the WOWA, finally computes the WOWA using the ww package. If there are no α-relevant arguments or an error is produced, the method returns NaN.

- `public void printStatus(PrintStream out, double num)`: Prints the status of all the arguments in the α-relevant argument set returned by `this.alphaRelevantArguments(num)`,calling the `printStatus(out)` method of each argument.

- The class also has get and set methods for some variables.

NormArgumentMap class

This class implements the norm argument map from which all the argumentation process will develop. It has a norm, the norm being discussed; two argument sets, one for arguments in favor of the norm and one for arguments against the norm; a spectrum, the spectrum of valid opinions and a real number alpha, the alpha used to distinguish α-relevant arguments from the others. To construct a new NormArgumentMap object you will need a norm and a spectrum. The importance function and the alpha are optional, if they are not introduced the default ones will be used. Arguments will be introduced via methods not in the creation of the NormArgumentMap.

Methods

- `public void addArgument(Argument a, Boolean positive)`: If positive is true and a is an argument which has the same spectrum as the NormArgumentMap, then a is added to the argument set of arguments in favor of the norm. Analogously if positive is false, the argument a is added to the argument set of arguments against the norm. If the argument a has not the same spectrum as the NormArgumentMap, nothing happens.

- `public void addArgument(String a, Boolean positive)`: This method has the same functionality as the previous, the only difference is that

in this case instead of receiving an argument, the method receives a string from which an argument will be created.

- `public double numberOpinions(Boolean relevant)`: If relevant is true returns the number of opinions of all α-relevant arguments (whether in favor or against the norm). Otherwise if relevant is false returns the number of opinions of all arguments.

- `public double minimumOpinions(double alpha)`: Returns the minimum number of opinions that an argument has to have to be considered an α-relevant argument, it is computed using the alpha passed as parameter.

- `public double minimumOpinions()`: Returns the same as the previous method but using the alpha in the NormArgumentMap instance.

- `public double support(double alpha)`: This method returns the support for the norm. For assessing the support, the method in the project is used. The alpha passed as a parameter is used as the α in the α-relevant argument choosing step.

- `public double support()`: As in the previous method, this one returns the norm support but using the alpha in the NormArgumentMap instance.

- `public void printStatus(PrintStream out)`: Using out, prints the status of the whole norm argument map, the data of the norm, the spectrum, the alpha and the results of support for each argument set and its arguments. Finally, also prints the support for the norm.

- `public static NormArgumentMap loadFile(File f)`: Loads the components of the norm argument map from the file f. For the structure that a norm argument map file has, head to the next section of this guide.

- The class also has get and set methods for some variables.

DefaultImportanceFunction class

This class has the implementation of the natural opinion importance function described in the project memo. To create a DefaultImportanceFunction instance you only need to define the spectrum of the function and a double containing the tolerance for the double values(as explained in methods). As an extension of the abstract class ImportanceFunction, this class has the methods defined in ImportanceFunction along with others:

Methods

- `public double compute(double x)`: As in the ImportanceFunction, this method receives a double and returns the value of the importance function in that point.

- `public double correct(double x)`: This method receives a double and corrects it with the tolerance. This is important because in some cases if arguments have some opinions of the highest value possible when aggregated the double can be higher than expected, hence be out of the spectrum. In this method this flaw is corrected.

DEFAULT class

The DEFAULT class only has three static methods and cannot be instantiated.

Methods

- `public static ImportanceFunction importanceFunction (Spectrum s, double tol)`: Returns an ImportanceFunction object which is the default importance function in the spectrum s and tolerance tol.
- `public static ImportanceFunction importanceFunction (Spectrum s)`: Returns an ImportanceFunction object which is the default importance function in the spectrum s and default tolerance 10^{-5}.
- `public static double alpha()`: Returns the default alpha which is 0.3.

ArgumentComparator class

This class implements a comparator to compare arguments, an argument will be greater than another argument if it has greater support. This is used to order arguments in q vector of the WOWA for the argument set support.

Methods

- `public int compare(Argument o1, Argument o2)`: Compares the two arguments comparing their support.

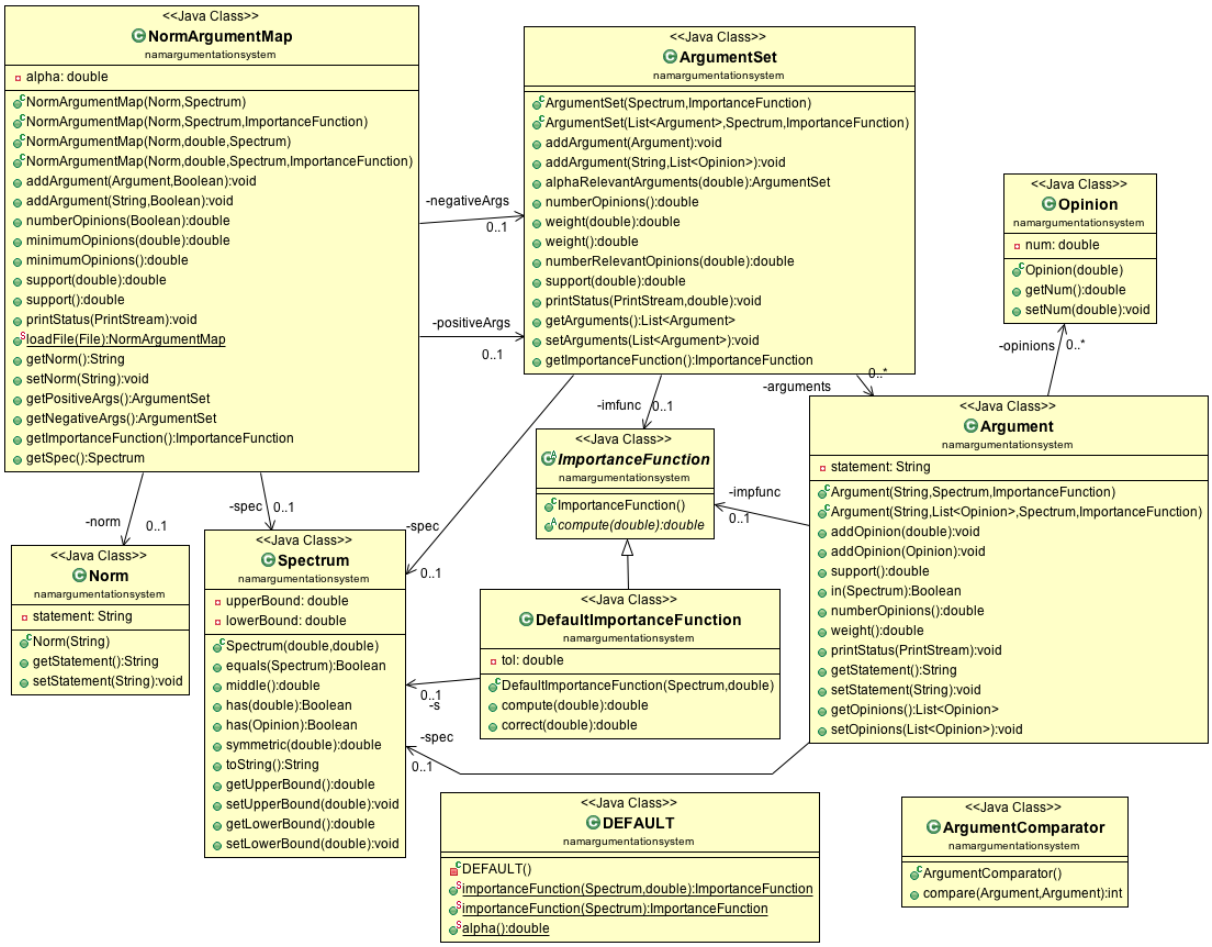

Figure B.1: The class diagram for the namargumentationsystem package.

B.2 The norm argument map file

The structure of a file loadable for the norm argument map must be as follows:

- The first line has all the data for structuring the argumentation in the norm argument map, it has to have 4 values separated by commas in the following order: norm statement, alpha, spectrum lower bound, spectrum upper bound. The norm statement and the spectrum bounds have to be introduced, whereas alpha can be left empty and the default alpha will be used.

- The next lines are for arguments and its opinions, each argument uses a different line, each argument line will have variable amounts of values separated by commas in the following order: argument type, argument statement, opinion1, opinion2, opinion3, etc. Argument type is either true (argument in favor of the norm) or false (argument agains the norm). For an argument to be considered valid only the type and the statement are necessary, because the number of opinions may vary from none to huge amounts.

Example. *Imagine we have a norm argument map debating the norm n, with the default alpha and the spectrum* $[1, 5]$*. And we have the positive arguments:*

- *posarg1 with opinions 1, 2, 4, 5*
- *posarg2 with opinions 4, 5, 4, 4, 5*

And the negative arguments:

- *negarg1 with opinions 1, 2, 1, 3*
- *negarg2 with opinions 1, 5, 4, 5*

Then the text in the file representing this norm argument map would be:

```
n,,1,5
true,posarg1, 1, 2, 4, 5
true,posarg2, 4, 5, 4, 4, 5
false,negarg1, 1, 2, 1, 3
false,negarg2, 1, 5, 4, 5
```

This example is saved in the file `testdata.txt` in the NormSupportEvaluation Java project, and can be loaded to the program to view the support results. The example in 5.2 is also in NormSupportEvaluation as file `example.txt`.

B.3 The NormSupportEvaluation Java project

The implementation in the NormSupportEvaluation Java project uses the package namargumentationsystem and the ww package done by V.Torra to execute a norm argument map evaluation in a command-line interface.

The FrontEnd.java file has all the methods necessary for the program and the command-line interface. To execute it (you will need to have Java installed) follow these steps:

- Open your command line interpreter in the NormSupportEvaluation directory.
- To execute the program enter: `java FrontEnd` (The program is already compiled, to compile it you could use `javac FrontEnd.java`).
- The menu of the program will appear, you can choose between two options, entering the data manually or from a file.
- If you decide to enter data manually, simply follow the steps shown in the screen.

- To load a file, simply type the file name (make sure the file is at the NormSupportEvaluation folder), you can use the test file provided in the package entering `testdata.txt`.
- When all the data is entered or loaded the program will show the results. If an error occurred it will also be shown.