Improving the PARMENIDES system with the use of Dungean formal models of Argumentation

Barend Poot

February 15, 2013

Student number: 3539784
Universiteit Utrecht
Begeleider en eerste beoordelaar: Henry Prakken
Tweede beoordelaar: Gerard Vreeswijk

Contents

1 Introduction

With the emergence of the internet, which has made an enormous amount of information available to the public, a revolution has taken place in the way we exchange information. This change is of great significance, especially in a democracy. In our modern democracy, representatives are chosen by means of elections and we trust these representatives to make the decisions we would make ourselves. However, there are people who feel that casting their vote periodically is not quite satisfactory; they want their own opinions and ideas to be part of the discussion. Formerly, citizens could take part in a political debate by writing letters or attending a public debate. Nonetheless, reaching a larger audience was not a simple task for regular citizens, since the best way to do this was using either the radio or television and the average person does not have access to these channels.

1.1 E-Democracy

The internet has brought about a great change in the manner in which citizens and their representatives communicate in our democracy. So much so, that it has spawned a new form of this political direction: E-democracy. E-democracy is a concept that creates many new opportunities. It should not only replace the current system with electronic counterparts, like electronic voting, but it should preferably create systems that introduce new functions. The internet already provides some of these functions but not in official government applications. For instance, social media is becoming an important platform for both the government and the public to spread information, opinions and ideas. Everyone with an internet connection can instantly reach an enormous audience by using Twitter, Facebook or other networking sites.

Even though it seems evident to use the internet for public discussions about government issues, such a platform does not currently exist in The Netherlands. The Dutch government provides several websites, such as `http://www.overheid.nl` and `http://www.officielebekendmakingen.nl`, but these sites only provide information in one direction; there is no interaction with the public. Creating an application or website that could function as a platform for discussion would be a logical next step in E-democracy. It combines the two main factors of both internet and democracy, exchanging information and involving the public with government decisions. Nevertheless, how could such a system be implemented? For an Artificial Intelligence student, this is a very interesting question. What is required for this task is some sort of system that is not only able to provide information, but can also interact with the user. This system should be able to support a natural debate, which can be done by using *argumentation*.

Argumentation or argumentation theory is the study of resolving a conflict of opinions with the use of logical reasoning. An important aspect to note here is that this form of logical reasoning must be *nonmontomic*. This means that new information (i.e. a new argument in a debate), has the ability to 'defeat'

earlier submitted information. This form of reasoning is also called *defeasible reasoning*. Evidently, in a natural debate new information can overrule previously introduced information, in the same way that new evidence in a court case can ultimately change the verdict. Therefore a program that is created to serve as a platform for discussion and to simulate a natural debate should be fundamentally supported by defeasible reasoning and argumentation theory. And since argumentation has been an important subfield of Artificial Intelligence for over twenty years, the creation of an E-democracy discussion application is not only socially relevant, but also relevant to my study.

1.2 Related Work

A project that provides a platform for discussion and does this by using argumentation theory, is the Parmenides system [2]. The Parmenides system was developed by several members of the Department of Computer Science at the University of Liverpool, UK. Among them are Katie Atkinson, Dan Cartwright, Trevor Bench-Capon and Adam Wyner. This system has been implemented as a website on which several topics can be discussed and allows the user to input their own information. Although this project looks very promising, it is not (yet) in actual use with the government of the United Kingdom.

The goal of this thesis will be to investigate the Parmenides system and if any limitations are found, to propose a solution or alternative system that might overcome these limitations. The question I will try to answer is the following:

"On which points can the Parmenides system be improved and how?"

Since there are multiple programs that bear the name Parmenides, the manner in which they work as well as their differences will be explained in section 2. Several basics of argumentation like argumentation schemes and argumentation frameworks are discussed there together with a brief description of another consultation tool, the SCT.

Section 3 will discuss the limitations found in both Parmenides programs, I will also propose some improvements that could be made within the program and finally propose a completely different system that should be able to overcome these limitations.

Section 4 is dedicated to the prototype programs created with the aim to improve on Parmenides. The structure of this program is explained by progressing through the program as a normal user would and reviewing every page.

Finally, section 5 will once more review both Parmenides and the prototype I propose; it will also discuss whether actual improvements have been made. Some possible options for future development of the new program will be discussed as well.

2 Parmenides and SCT

In this section we will look at the inner workings of the PARMENIDES system. To understand what aspects of this program can be improved we must first

understand the program itself.

Atkinson et al. present the PARMENIDES system as a prototype tool used to gather and analyse public opinions regarding the justification of policies. The final goal is to make a general dynamic program that can register opinions and new input from the general public, whilst also providing the public with information on the topic.

Before we take a closer look at the PARMENIDES system, we must differentiate between the different versions. There are two versions with the name Parmenides and one similar, but fundamentally different system named the Structured online Consultation Tool (SCT). The SCT is currently being developed by the IMPACT group[1].

2.1 Prototype PARMENIDES

PARMENIDES is the first version of the two Parmenides systems. I will from now on refer to this system as PARMENIDES or Proto Parmenides, as Cartwright does in [1].This system is implemented by a PHP-made website which at the time of writing can be found at the following address: `http://cgi.csc.liv.ac.uk/~katie/Parmenides1.html`. The program is explained in [2], however, I will give a brief summary here.

Atkinson first presents the aim of the program, PARMENIDES should justify an action or course of action from one party to another. In this case, the proponent of the action is the government. The program discusses the following topic:

"Is Invasion of Iraq Justified?".

This is done by using a particular Argument Scheme (AS), in this case the argument scheme of Practical Reasoning, it is an extension of Douglas Walton's[7] *sufficient condition scheme for practical reasoning*. This argument scheme (AS1) is stated as follows:

- In the current circumstances R

- We should perform action A

- To bring about new circumstances S

- Which will realize goal G

- Which will promote value V.

In an earlier article [3], a list of attacks against this scheme was formulated. 15 different types of attack were identified (see table 1). However, Atkinson et al. state that if the proponent of the position only produces sound and well-formed arguments (although the terms 'soundness' and 'well-formed' are not exactly defined in [2]), some of these attacks can be ignored. Attack number 3

[1]More information here: `http://www.policy-impact.eu/`

should be omitted for example, because this attack questions the consequenses of the action, which we assume would logically follow if the argument is sound. Attacks 6,7,9 and 11 are all about alternative actions and the user has the option to propose alternative actions and arguments at a number of points in the program. These attacks are addressed by providing that option. Atkinson et al. chose to not use attack 10 because it does not really attack the topic. And finally we can ignore 12, 13 and 14, as we assume that the states of affairs and actions described are at the very least possible. In this way, only 6 remain. The remaining attacks are numbers 1,2,4,5,8 and 15 from the following table:

Attack	Description
1	Disagree with the description of the current situation
2	Disagree with the consequences of the proposed action
3	Disagree that the desired features are part of the consequences
4	Disagree that these features promote the desired value
5	Believe the consequences can be realized by some alternative action
6	Believe the desired features can be realized through some alternative action
7	Believe that an alternative action realizes the desired value
8	Believe the action has undesirable side effects which demote the desired value
9	Believe the action has undesirable side effects which demote some other value
10	Agree that the action should be performed, but for different reasons
11	Believe the action will preclude some more desirable action
12	Believe the action is impossible
13	Believe the circumstances or consequences as described are not possible
14	Believe the desired features cannot be realized
15	Disagree that the desired value is worth promoting

Table 1: A list of all 15 attacks

Using these six remaining attacks, the program lets the user follow a standard path through questions and statements. On each page the user has the option to agree or disagree with a statement, and on particular pages the user can input his or her own opinion.

On the first page, the user can enter his or her name, and check the main topic of discussion. After this is done, the next screen presents a structured statement of the position to be considered, in the form of AS1. The user must decide on this page if he or she agrees or disagrees with the discussion topic. If the 'Agree' option is selected, the program is terminated immediately. If not, the user is taken through a series of pages. Each page represents one of the remaining 6 attacks presented above. The user can input his or her opinion here, until the final page has been reached. The final page is a summary page: All the collected information will be shown here. Finally, the user can choose to start again at the beginning, or continue the discussion by explicitly stating their own views on the topic. If the user chooses the latter, his or her own views can be recorded. These can possibly be added to the program by a human operator.

One aspect that all consultation tools that are reviewed in this thesis have in common, is that they require the use of an human administrator to process user-submitted information. One of the advantages of this system is that everyone can use it and no prior knowledge of argumentation theory is needed. However, as the system is used by the general public, the input is too diverse for a computer to handle. As such, a choice must be made. Either a human operator sorts all input, converts it into general statements and enters the general statement into the correct place in the program or all input is instantly allowed into the program.

Both sides bring advantages and disadvantages. While using a human administrator will result in a lot more structure and a lot less nonsense, it also carries a great risk. If an operator is not completely objective he or she might choose to withhold certain arguments, which will result in a biased discussion. A discussion without any supervision however will likely be filled with irrelevant arguments submitted either accidentally or on purpose by unwilling users. This issue will not be discussed further in this thesis. Nevertheless, it is an important aspect that must be considered if the online consultation tools will actually be used by a government.

PARMENIDES is made so that it is easily accessible to the public, no prior knowledge of argumentation theory or argumentation schemes is required to use the website.

2.2 Parmenides

Parmenides is an improved version of the prototype PARMENIDES. This system is also implemented as a website. At the time of writing, the site can be found here: `http://cgi.csc.liv.ac.uk/~parmenides/`. I will refer to this program as Parmenides.

Dan Cartwright[2] states that Parmenides has 3 main functions. The first function is to provide support for the creation of arguments concerning recent political issues. This is done by creating a platform with the essential capacities for providing such support. The second function is to give the general public an opportunity to reply to statements and decisions put forward by the government. The user can enter his own input, but will also learn more about the particular issue by reading and participating in the debate. The third and final function is the evaluation of the information that the government receives. Everything the general public submits will be used to create an image of the general opinion of the users. In Parmenides' case, this is done using a graphical interface.

The main difference between the two systems lies in the diversity. The second version of Parmenides has a larger number of options, the first thing you will notice is the variety of topics to choose from. Parmenides now supports a so-called *Debate Creator*, which enables an administrator to easily generate a topic in the same framework. More importantly however, the new system also

[2]Cartwright states these functions and the main differences between the Prototype PARMENIDES and Parmenides in this [4] article

supports new argumentation schemes. Three other Walton [7] schemes have been added to the program:

- *Argument from expert opinion*

- *Argument from position to know*

- *Argument from correlation to cause*

As only one argumentation scheme was used in PARMENIDES, users were limited in the variety of input they could submit. By adding more argumentation schemes, Parmenides has greatly improved the way in which users can express themselves. Of course, this change also improves the amount and diversity of information that the proponent (e.g. the government) receives.

In terms of analysis, the program has also improved: Parmenides uses *Argumentation Frameworks* [8] (AF). An argumentation framework consists of a tuple:

$$AF = \langle AR, Attacks \rangle$$

AR is the set of all arguments in the framework. *Attacks* is a binary relation on AR.

$$Attacks \subseteq AR \times AR$$

In practice, this means that if (A, B) is an element of *Attacks*, A will attack B, defeating the argument B in the process. Using an argumentation framework, a network of nodes (arguments) and directed edges (attack relations) can be created, we call such a network a *Dung Graph*. Parmenides uses these argumentation frameworks to analyse the input received. Depending on new user

Figure 1: A Dung graph with three arguments and two attack relations

information, attack relations may be discarded. Which relation will be discarded can be calculated by using so-called Acceptability Semantics. In this process two different phases can be distinguished. I will briefly explain these acceptability semantics here by first giving some definitions and then explaining how we can use these terms to decide whether or not certain arguments are 'acceptable'. See [9] for a more in-depth explanation.

Let $B \subseteq AR$

The set B is *conflict-free* iff there exists no arguments $X_i, X_j \in B$ such that $(X_i, X_j) \in Attacks$

The set B *defends* an argument iff for any argument X_i that is attacked by an argument $X_j \in AR$, an argument $X_k \in B$ exists such that $(X_k, X_j) \in Attacks$

Futhermore, a *grounded extension* set is the set B that is conflict-free, defends all its members, contains all arguments it defends and is the smallest set that has these properties. The grounded extension is a unique extension of the AF. By checking if an argument belongs to the grounded extension set, it is possible to see if an argument is *justified*, *overruled* or *defensible*.

An argument A is:

Justified if and only if A is *in* in the grounded extension set.

Overruled if and only if A is *out* in the grounded extension set.

Defensible if and only if A is undecided in the grounded extension set.

In the first phase, arguments are checked for acceptability by looking if they belong to the grounded extension set. This means that any argument that is *Justified* is acceptable. In the second phase, certain attack relations may be discarded according to preference. A difference between attack and defeat relations must be noted here. The two terms are often used as synonyms, but there is a crucial difference if preference is used. If there is a significant number of users that agreed with a statement A compared to a small amount of users that agreed to statement B, a certain preference for A is developed. This preference can be used to update the Dung Graph and certain attack relations might be discarded if the argument that is being attacked has preference.

In figure 1, three arguments and two attack relations are defined. Arg2 attacks Arg1, however, because nothing is attacking Arg3 and Arg2 is attacked by Arg3. We can ignore the attack relation between Arg2 and Arg1. Basically this means that Arg3 and Arg1 are *Justified*. The attack relation is not discarded, but Arg2 is simply not in the grounded extension set. There is no need to look at preference in this example.

Now let us imagine a situation with two arguments and two attack relations, both attacking eachother. If one of these arguments is preferred above the other, this argument will defeat the other, discarding the opposing attack relation in the process.

Using this method, each time a new argument is added, the argumentation framework and the corresponding Dung graph can be modified, which will result in a dynamic discussion.

2.3 Structured Online Consultation Tool

The IMPACT group summarizes itself as the Integrated Method for Policy Making Using Argument Modelling and Computer Assisted Text Analysis. Among

many other things, they are working on the SCT, an advanced, although fundamentally different version of Parmenides. Since this project is still in development and since it differs from both versions of Parmenides, we will just take a look at the obvious differences between the two. A thorough research is not needed because in this thesis we only look to improve the Parmenides systems.

According to the developers, the SCT differs from previous consultation tools because it uses *"interconnected, expressive and formalised argumentation schemes"* [13]. Both versions of Parmenides use argumentation schemes, the second Parmenides system even has the option to create another debate about the same subject using another argumentation scheme. However, this still differs from the way the SCT implements argumentation schemes. As stated above, the SCT uses interconnected argumentation schemes. Basically, this means that the SCT applies multiple argumentation schemes in the same debate. The so-called Practical Reasoning scheme (AS1) is always a central part of the debate, because all policy proposals are based upon a justification of what to do on a specific issue. But other schemes are used in the same debate: Expert Opinion, Ad Hominem and Argument from Analogy, for example. By connecting different argumentation schemes, the user and the developer both have more freedom in respectively receiving and supplying information. The larger the network of different schemes, the more options are avaliable for both user and developer.

At the time of writing, the final prototype of all argumentation tools made by the IMPACT Project group can be found at the following address: `http://impact.uid.com:8080/impact/#`. The project has just ended after having run for 3 full years. Future planning for the development of the SCT includes adding more schemes.

3 Limitations of the Parmenides system.

After having discussed the most prominent existing online consultation tools, we will now check if there are areas in which they can be improved. As we have seen before, Cartwright [4] states that the Parmenides program must fulfil three major functions. In my opinion each online consultation tool should possess at least these three functions, albeit implemented in different ways. We will look at both versions of Parmenides and check if the implementation of any of the three functions can be improved. First of all, I will state the three major functions once more.

1. Providing a platform for the discussion to take place and to give the public a clear overview of the discussion that has taken place regarding the decision that was made.

2. Allowing the public to respond to the discussion and eventually input new arguments.

3. Using the data that was gathered to analyse a general opinion of the public and possibly alter the discussion or decision according to the public's response.

Evidently, both versions of Parmenides try to fulfil these functions. But even so, I will show that there is room for improvement.

Futhermore it must be noted that both

3.1 PARMENIDES limitations

PARMENIDES is a prototypical program, so evidently there should be room for improvement. Any deficiencies can be discovered by checking how the functions formulated above are implemented in this program. The first function requires that the program is a proper platform to discuss the topic in question. The user should also have a clear overview of the entire discussion that preceded the decision that was made. Of course, this means that all the information should be accessible. Personally, I believe this is one of the most important functions of the system. The entire discussion that took place in government is always made available on government websites, and it should be the same on every online consultation tool. It is true that some parts of the discussion are of greater importance or more interesting than others, nevertheless everything should be available for viewing and discussing.

PARMENIDES does not quite live up to this requirement, even though it has only one topic that is discussed: *"Is Invasion of Iraq Justified?"*, the use of the practical reasoning scheme restricts this program from displaying all information of the topic. Although argumentation schemes are a natural and rational manner of presenting and giving justification to an argument, they do not simulate a debate. A debate must be interactive. And even though the opponent (the user) can agree or disagree with every argument, these responses have little effect on the debate. The user basically follows the same path everytime the program is executed, a few exceptions notwithstanding, e.g. when he/she immediately chooses for the option "Agree" at the beginning. Because the entire path must be followed, the user has to read and process all the information entered in the program. If every piece of information available on the topic were entered, this program would hardly be an alternative to reading an entire dossier about the discussion. As the use of argumentation schemes restricts the user in choosing what information (not) to read, it indirectly restricts the amount of information that can be entered in the program. The expressiveness of argumentation schemes is insufficient to present the entire discussion in a clear and accessible way. An optimal combination must be found of presentation all information and giving the user the choice on what to read. In PARMENIDES, with the restricting use of the practical reasoning scheme, this combination is far from optimal.

The second requirement is that the program provides the user with the option to respond to arguments. The user should also be able to add new arguments that may be included in the system later. It seems PARMENIDES meets the requirement of enabling user input. At several stages in the program, the user is prompted to add his or her opinion about the matter. And after completion, in the summary phase, the user can even choose to "continue the discussion by explicitly stating your own opinions." If this is done, a lot of extra information is

offered for the administrator to process. Fortunately, because it is optional, only users who actually want to do this (and hopefully have some coherent points) will provide extra input. The only remark I would like to make here is that the amount of information that must be filtered by the human administrator will, with a group of users of considerable size, take on large proportions fairly quickly.

The third and final function of the program should be to analyse and structure all output. The system should also be able to adapt according to the this output. Arguments may be added or removed, but only the administrator should have the authorisation to do so. Analysing the data can be done in many ways. However, since there is no documentation on the analysis of user data by PARMENIDES, I assume this is simply not implemented in this prototype program.

3.2 Parmenides improvements

Parmenides is the improved version of PARMENIDES and should no longer be called a prototype. In the best case, all issues raised in section 3.2 should be resolved. Once again, we will check the functions above against this version of Parmenides, and see what improvements have been made. The biggest flaw we found in PARMENIDES was that the use of argumentation schemes was restrictive. Parmenides has improved on this. As I stated before, Parmenides uses more than just one argumentation scheme. The debate creator is the part of the program that is able to quickly create a new debate with several argumentation schemes to choose from. Using these different argumentation schemes, topics can be divided into several smaller topics with different argumentation schemes. In this way, the user can choose to participate only in the discussions he or she is actually interested in. This wil presumably result in not only a better accessibility for the user, but also more intelligent and relevant input for the administrator to process. All this while all information remains accessible. Even though improvement has been made by adding more different schemes, the use of argumentation schemes itself remains restrictive. As such, adding more schemes is not sufficient to overcome this limitation.

The second issue in PARMENIDES was the user-submitted information. Although this is implemented in a good way, the amount of information has the potential to rise to enormous proportions, even more so if a large number of users explicitly state their opinions at the end. Not only is this a lot of information, it is also submitted at the same stage in the program. The administrator has to process the information, formulate a general argument representing all relevant input and finally insert this new argument in the correct scheme and place.

The solution possible in Parmenides, creating different subtopics with various argumentation schemes, may also be conducive to solving the problem of processing these substantial amounts of user input. Input will not only be presumably more relevant, like I stated above, but also already in the right place. If the user submits information, he does so at a specific phase in the program. If this phase is recorded together with the new input, the information

can automatically be sorted, saving the administrator a significant amount of work. Although dividing topics into smaller topics with different schemes will probably help the administrator in terms of getting better and more relevant input, it will also increase the amount of work needed for creating the different subtopics. And the lack of choice that the user has while using the program is still a limitation. The user might a greater choice of subtopics, but the interactivity of the debate itself has not increased. The optimal combination of preserving information and giving the user more accessibility.

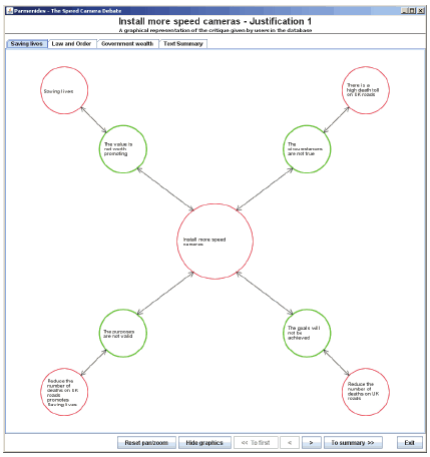

Figure 2: The Java-based analysis tool of Parmenides

Unlike PARMENIDES, the Parmenides program offers sufficient documentation about the method of analysing output data. Cartwright et al. discuss the so-called *"Analysis tools"* in [6]. Note that none of these analysis tools are available to the users, only the administrators can access them. The analysis tools are Java-based and generate argumentation frameworks in accordance with the input entered. First a visual representation of the argumentation framework is generated. This is basically a Dung Graph where all arguments are displayed as nodes and all attack relations as arrows in a directed graph. All arguments are delineated either red or green, depending if the majority of votes they received were "Disagree"or "Agree", respectively. Some nodes display a + sign inside, this can be clicked to expand the branch. Supporting evidence and critical questions concerning this node will then be displayed. Another feature of this analysis tool is that the percentage of respondents that agree or disagree will be shown if the mouse is moved over the node of the particular statement.

This seems to be a proper way of combining existing arguments with the input given by respondents. And because all information is so easily accessible, it is simple to see what arguments are most commonly agreed with. For a single debate, several of such Dung Graphs are built, different graphs for different values. The Java-based application shows different tabs where each of these graphs can be accessed. Parmenides handles the analysis quite well, all information is

recorded and displayed in a simple and intuitive way and supporting evidence is very accessible.

3.3 A proposal

Although some possible solutions have been presented above, in this section I will propose a fundamentally different system that deals with these limitations in an alternative way.

Although Parmenides overcomes many of the limitations encountered in section 2.1, an improved balance between preservation of information and an accessible system has still not been achieved. The user should be able to complete the program while only accessing the information he or she is actually interested in. In this way, using the website will not only be a more enjoyable experience, but it will probably also yield more relevant information. I believe the restriction on information originates mainly from the use of argumentation schemes, because they follow a specific pattern with certain goals, values and results. And this is where the solution can be found.

I devised and constructed a prototype which does not use these argumentation schemes. It will be discussed and reviewed in the next section. This prototypical system is called Heraclitus.

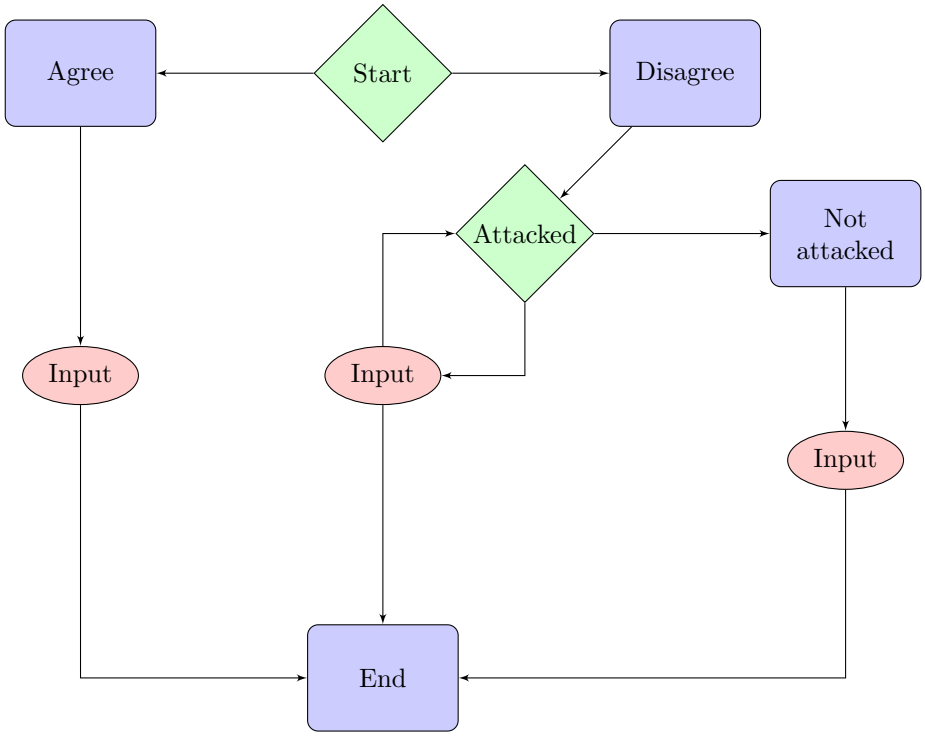

Figure 3: A flowchart representing the structure of the Heraclitus program.

4 Heraclitus

Heraclitus is a prototype program created with the aim to improve on the Parmenides system. Both systems have the same goal: to realize the three functions stated in section 3, though they do so in fundamentally different manners.

The Heraclitus program differs from the programs discussed above in that it does not use any kind of argumentation scheme. As argumentation schemes have a restrictive nature, another manner of progressing through a discussion must be found. This is where argumentation frameworks come in to play. As discussed in section 2.2, these argumentation frameworks can be respresented by Dung graphs: a network of arguments represented by nodes and attack relations represented by edges. A Dung graph is much like an actual discussion, arguments going back and forth, each one attacking the other. Even though this could work counter-productive an AF has the abilty to capture any argument as long as it is related to the topic of discussion. And because arguments must always attack an existing argument, it also captures the chronological layout of the debate. Because this layout is captured, the generated debate wil have a natural 'feel'.

To keep this natural feel, Heraclitus retains the structure of the Dung graph. The user travels from node to node via the attack relations. In this manner, each node can be reached, but does not have to be reached. The user only receives the information in which he or she is actually interested.

This program is designed as an framework for existing discussions. In the first place, an administrator must enter several arguments and their attack relations into the database by using the *Administrator Panel*. After this has been done, users can participate and enter their own arguments, thereby expanding the debate in the process.

4.1 Start phase

The Heraclitus program, very much like Parmenides, has been implemented by a PHP-website with a database in MySQL which can be found at this address: `http://www.phil.uu.nl/~poot`. The website has been built so that it can provide a general framework for any topic. Changing the topic is as simple as importing another database with the same structure. The program has a simple structure that is shown by the flowchart in Figure 3. Of course this is not all there is to the Heraclitus program. For simplicity's sake, the connection to the MySQL database and the Administration Panel, as well as all PHP process pages have been omitted. The flowchart only shows what the typical user can access.

Each node in the flowchart represents a page on the website, except for the red oval nodes. These stand for the optional decision to add input and can be used on the pages of the nodes that point towards them. The green diamond-shaped nodes represent pages where the user has to make a decision. In each of these nodes information is added to the user's personal database; this database will be discussed later. The blue square nodes represent pages where the user

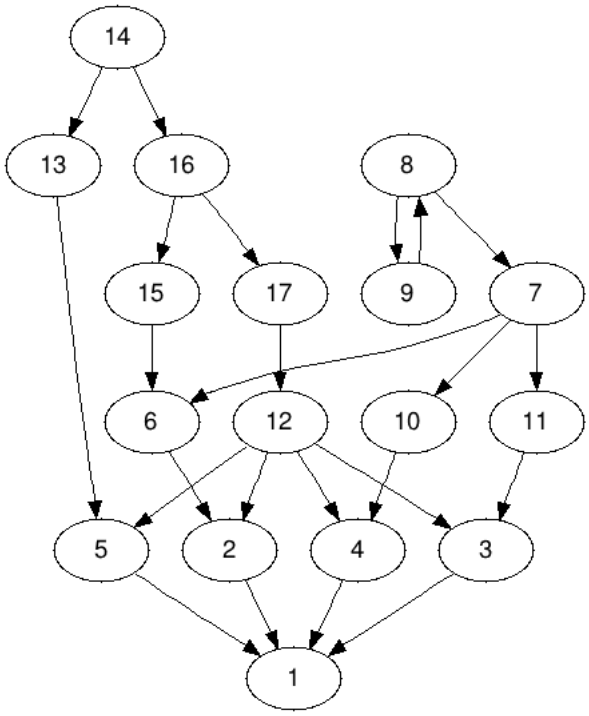

Figure 4: A Dung graph of the mock debate currently used in Heraclitus for development

mainly receives information before progressing to another phase.

The user starts on the 'Start' node, a decision should be made here to either agree or disagree with the topic presented on the start page. A mock[3] debate has been set up to show how exactly this program works, the Dung graph of this debate is shown in Figure 4. The topic is the following:

"All nuclear facilities in the Netherlands should be removed"

On this page the user is introduced to the program, some information is stated above the topic as can be seen in Figure 5. The user is also asked to enter his or her full name and is notified that no personal information will be recorded, this will be made clear when the databases are discussed. Notice the section on the left-hand side of the screen, this shows what phase the user is currently in. Not all different phases are represented by a separate node, the nodes 'attacked' and 'not attacked' correspond with a single phase: 'Discussion'.

The section in the bottom right-hand corner provides the user with some extra information. The notification that no personal information will be recorded

[3]Please note that the arguments in this debate have not been checked for vadility. They may be untrue or imply false consequenses. This debate is merely a placeholder.

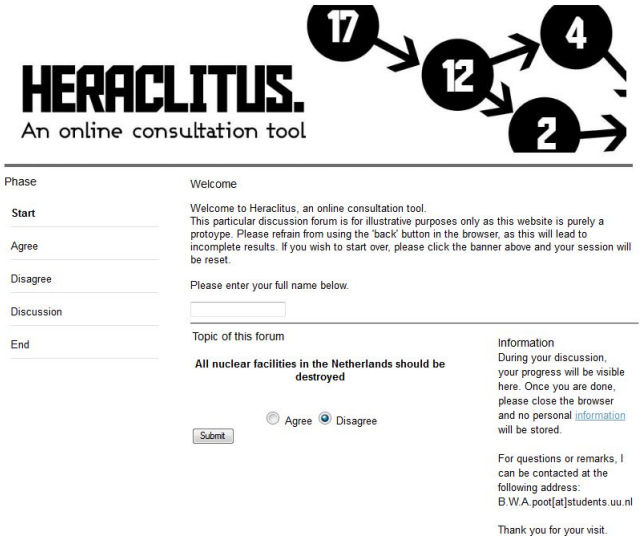

Figure 5: A screengrab showing the Start Phase

is stated here, as we can see in Figure 5. This section will be of more importance on later pages of the program.

4.2 Agree phase

If the user agrees with the initial statement, he or she will be taken to this screen, which is quite straightforward. The user has 2 choices here. Either just end the program, or provide some input of their own and fill in the form. This form corresponds with attack 10 as formulated in Table 1: "I agree that the action should be performed, but for different reasons."

As can be seen in Figure 6, the bottom right-hand section has now been updated and shows the progress the user has made so far. Upon pressing the 'Submit' button the user will be taken to the 'End' phase, which we will discuss after what happens if the respondent chose to disagree with the topic.

4.3 Disagree phase

In the disagree phase the user can choose from all arguments that attack the initial topic. As can be seen in Figure 4, there are currently four different arguments attacking argument number 1, which is the topic of the debate. Because there are four arguments attacking the topic, the user will have four options to choose from on this particular page. However, because all options that can be made here will take the user to the next phase, this page is not represented by a decision node. The decision that is made here only influences the personal table of the user and not the flow of the program.

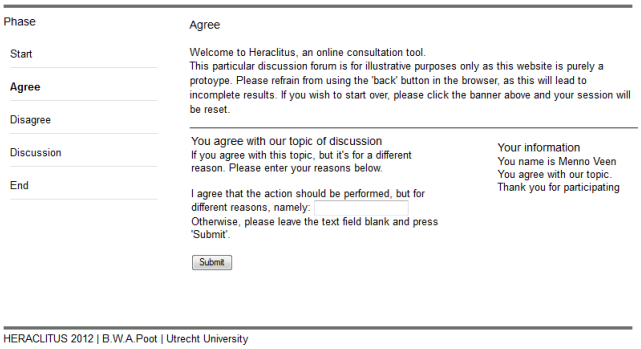

Figure 6: A screengrab showing the Agree Phase

Unlike all other pages where users can choose to attack arguments, this page does not allow the user to submit information. This is purely a design choice, all user-added arguments should be specific and should be entered in the correct place in the graph. If an option would be available for the users to input arguments on this page, I fear many people would enter their argument immediately and not participate in the rest of the discussion. This can be prevented by creating a select number of general arguments upon the creation of the debate and only letting these arguments directly attack the topic. This way, a user will have to use the program to move through the graph until he or she is in a proper place to insert an argument, which will result in a better and more rational discussion.

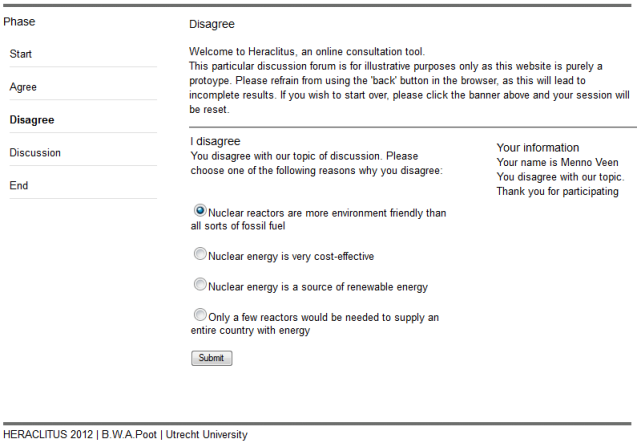

Figure 7: A screengrab showing the Disagree Phase

4.4 Discussion phase

The Discussion phase is very similar to the Disagree phase, although there are some crucial differences. The first thing to be noticed in the discussion phase is the option to end the program at any time by submitting an argument instead of picking one out of the list. This option is available for users who disagree with the statement on the page and do not agree with any of the provided attacks either.

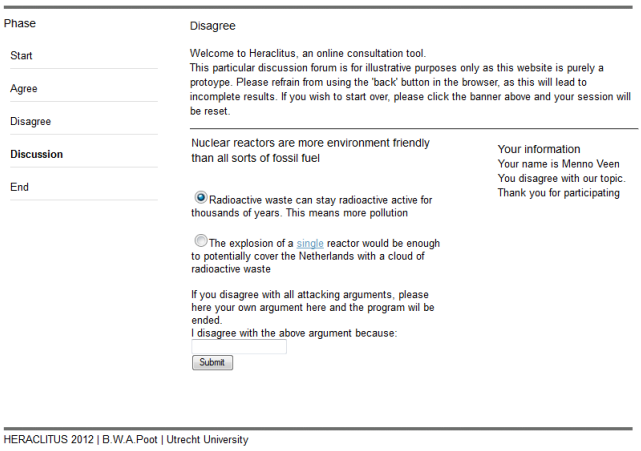

Figure 8: A screengrab showing the Discussion Phase

This phase is where the user will spend most of his or her time. Each time the user chooses an argument, the program checks if there are any other arguments that attack the chosen argument. If one or more of these arguments exist, the program will stay in this phase. Otherwise, the user will be taken to a page stating that no more attacking arguments remain in the database. Either an new argument can be submitted here or the user can choose to move on to the 'End phase'. In Figure 3 this page is represented by the nodes 'attacked' and 'not attacked'.

4.5 End phase

Finally the user will arrive at the last phase. This stage can be reached by any of the optional 'input' nodes, or otherwise through the 'Not reached' and 'Agree' page if no input is submitted. All arguments that the user has visited while traversing through the graph are registered and printed in the bottom right-hand section. If a personal argument has been submitted, it is also printed in this section. The user will also see the entire Dung graph of the debate with all nodes that were visited delineated by a red line. To keep the graph somewhat smaller, nodes only show an integer as a label and sthe corresponding argument is shown if the user hovers the cursor over the node.

This page provides a clear summary of the session that was just finished. A personal table containing all relevant information is already stored in the database, so two options are left. The user can either click the link which will destroy the current session and allows him or her to try again, or simply the browser.

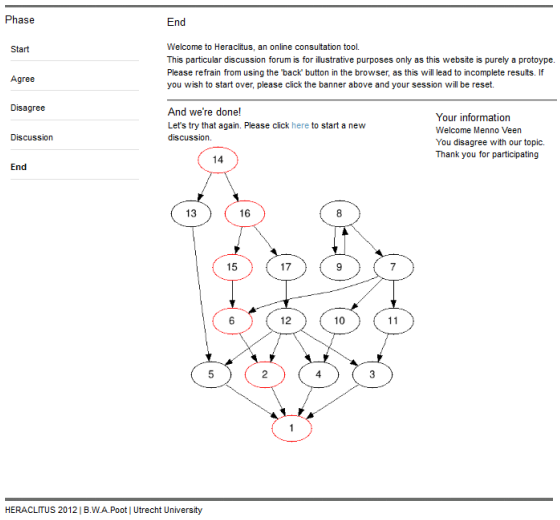

Figure 9: A screengrab showing the End Phase.

4.6 Databases

Heraclitus employs a MySQL server which uses two separate Databases. The first database is called *framework_db* and contains three different tables. *Arguments* is a table that has three columns: `id`, `argument` and `reached`. The column `id` contains an integer which corresponds with the number presented in the Dung graph in Figure 4. The column `argument` contains a string with the actual argument. And finally the column `attacked` contains a boolean value that shows the value `True` if the argument is attacked by another argument.

The two remaining tables are *defDirect* and *deftedDirect*. These tables are so-called combination tables and they both have 2 columns. Each table has a column called `id` and another one named `defeats` and `defeatedby` respectively. Each row in these tables represents an attack relation, in *defDirect* the first column lists the identity of the attacking nodes and the second columns lists the identity of the nodes being attacked. The table *deftedDirect* does exactly the oppostite.

I chose to make separate tables to record each relation because each node can have a different number of attack relations and recording a different number of relations for each argument in a single table is very awkward to implement. In this manner each relation is recorded in a single row in the combination tables.

Therefore *Arguments* and one of the combination tables can be combined and the system can cope with any number of attack relations on a single node.

The second database is *persons_db*. Every time the 'Submit' button the first page is pressed, a new personal table is created in this database. Each table is named *Table_X*, *X* being an integer that increases by one with every new user. The table contains the following information: All arguments that the user encountered, whether the user agreed or disagreed and finally any input that the user might have submitted.

This information is used to display the graph at the summary screen and in the future will be used for the analysis process.

4.7 Analysis

As stated in section 3.2,Parmenides also uses argumentation frameworks for the analysis of user output. Although analysis has not yet been implemented in this prototype consultation tool, my proposal is to analyse the output in very much the same manner.

Firstly, updating the graph in accordance with user-submitted input using preference is very important. If certain arguments seem of greater importance to the general user (i.e. are more often selected from the list of attacking arguments), the possiblity exists that certain attack relations will be discarded, just as in the original Parmenides system.

After a set number of visitors, all input will be checked. If any coherent, interesting and relevant arguments were submitted, they will be added to the database. Futhermore, all nodes will be updated with a 'preference-factor'. If this factor is high, this node has a greater chance of defeating other arguments and attack relations against it if a conflict may arise. Some attack relations may be discarded. This removal is, just as in Parmenides, entirely dependent on the Acceptibility semantics of Dungean abstract argumentation frameworks as explained in section 2.2.

Unfortunately, these Acceptability semantics are not implemented by Heraclitus. There are however several tools[4] available online which can perform these operations. Every time a change has been made to the database, and therefore the graph, the graph should be entered into one of these tools and be updated accordingly

4.8 Comparing Heraclitus with Parmenides

Now that the difference between Heraclitus and Parmenides is clear, we can compare the two systems against eachother. Heraclitus avoids using argumentation schemes in order to achieve more freedom in the discussion. Argumentation schemes however are a natural and rational way of presenting and justifying an

[4]One of these is OVA-Gen, which can be found here `http://ova.computing.dundee.ac.uk/ova-gen/`

argument. As such, new argument that are added in an existing argumentation scheme can be considered 'valid' if they don't directly contradict with the arguments in the scheme.

Because no argumentation schemes are used in Heraclitus, the structure they impose in the debate is lost. Even though the sole use of argumentation frameworks increases the freedom of the user, it is also harder to see if arguments are valid because they can not be checked against an argumentation scheme. Only using a Dung Graph as structure may result in a chaotic discussion with irrelevant arguments.

An advantage of the argumentation frameworks is that both users that agree or disagree can participate in the discussion, because arguments for both sides in the discussion are recorded. In conclusion, the restrictiveness of argumentation schemes is both an advantage and a disadvantage and the same goes for the freedom of using argumentation frameworks.

5 Conclusion

In this thesis a prototypical program is presented that aims to improve existing consultation tools. Even though governments do not yet use these consultation tools, I believe they will make up an important part of future E-Democracy and are therefore of great importance. After reviewing the existing Parmenides system which consists of two separate programs, which I refer to as PARMENIDES and Parmenides, certain limitations to these consultation tools are discovered. The main limitation that was found in section 3 was that not all available information could be displayed whilst also keeping the program accessible for the user, even though these two aspects should be fundamental to all consultation tools. While I also proposed a solution to this problem that could be implemented in the Parmenides program, I believe that the best combination of these two factors can be reached by creating a fundamentally different system. Therefore I have constructed a prototype. The structure of this program, which I named Heraclitus, is based entirely on argumentation frameworks as presented by Dung.

5.1 Discussion

The fact that the user is able to traverse through a so-called Dung graph produces several advantages. Not only has the user an enormous increase in freedom, it also gives a more natural feel to the discussion because arguments keep going back and forth. New input can be added at almost every stage of the program. This promotes interactivity and keeps the database of arguments dynamic so the discussion can be visited multiple times by the same users.

But as can be seen in section 4.8, using AF's also creates some serious disadvantages. Completely abandoning argumentation schemes corresponds with losing structure in the debate.

However, I believe that in a developing field like E-democracy, different viewpoints are important and further testing should show if these programs might ever be used as government consultation tools.

5.2 Future development

For future development, certain aspects come to mind. One of the major limitations that Heraclitus has in common with Parmenides, is that the program is almost immediately terminated when the user chooses to agree with the topic. An easily implemented and viable option to remove this limitation in Heraclitus is to simply insert another argument into the database that states the exact opposite as the current topic. This argument can also be used as a starting point and treated in the exact same way as the current topic. That way the graph simply has two starting points instead of the one point it has now. Because the arguments are going back and forth, the two starting topics will blend into the same graph and all users, whether they agree or disagree, can participate.

Another improvement could be implemented at the End phase, an option could be added to allow the user to add priority to certain arguments he or she finds significantly more important than other arguments. This would allow for a more detailed graph and better analysis of the submitted information.

Futhermore an improvement that PARMENIDES already has implemented can be made, allowing the user to click on certain statements. If a statement is clicked upon, it will yield background information and if applicable supporting evidence. These improvements are all easy to implement in Heraclitus and should definitely be added in a final version.

Acknowledgements

I would like to thank Henry Prakken, Gerard Vreeswijk, Lies Dijkstra, Vincent Koops and Tom Jager for their supervision, reviews, suggestions and expert opinions.

References

[1] D. Cartwright and K. Atkinson Political engagement through tools for argumentation. In *A. Hunter (editor): Computational Models of Argument, Proceedings of the Second International Conference on Computational Models of Argument (COMMA 2008)*, pages 116-127. Toulouse, France.

[2] K. Atkinson, T. Bench-Capon and P. McBurney PARMENIDES: facilitating deliberation in democracies. *Artificial Intelligence and Law*, Special Issue on eDemocracy, edited by T. van Engers and A. Macintosh. Vol. 14 (4), pp. 261-275.

[3] K.Greenwood, T. Bench-Capon and Peter McBurney. Structuring dialogue between the People and their representatives. In R. Traunmüller (editor): *Electronic Government*, Lecture Notes in Computer Science (LNCS) 2739, , pp. 55-62 Springer, Berlin, Germany. *Second International Conference on eGovernment* (EGOV 2003), DEXA 2003, Prague, Czech Republic.

[4] D. Cartwright, K. Atkinson and T. Bench-Capon Parmenides: Structured argument creation, response, and analysis. The University of Liverpool, UK.

[5] D. Cartwright and K. Atkinson. Using computational argumentation to support e-participation *IEEE Intelligent Systems. Special Issue on Transforming E-government and E-participation.* Vol. 24(5), pages 42-52.

[6] D. Cartwright, K. Atkinson, and T. Bench-Capon. Supporting argument in e-democracy. In *A. Prosser (editor): Proceedings of the Third Conference on Electronic Democracy (EDEM 2009)*, pages 151-160. Vienna, Austria.

[7] D.N. Walton. Argumentation Schemes for Presumptive Reasoning Lawrence Erlbaum Associates, 1996.

[8] D. Phan Minh. On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n person games. *Artificial intelligence* 77.2 (1995): 321-357.

[9] H. Prakken. An overview of formal models of argumentation and their application in philosophy. *Studies in logic* Vol 4, no 1 (2011): 65-86.

[10] T. Bench-Capon, H. Prakken and W. Visser. Argument schemes for two-phase democratic deliberation. *Proceedings of the 13th International Conference on Artificial Intelligence and Law*, Pittsburgh, PA, 2011. New York: ACM Press 2011, 21-30

[11] H. Prakken. Formalising a legal opinion on a legislative proposal in the ASPIC+ framework. In B. Schafer (ed.), *Legal Knowledge and Information Systems. JURIX 2012: The Twenty-fifth Annual Conference*, 119-128. Amsterdam etc, IOS Press (2012).

[12] H. Prakken and G. Vreeswijk, Logics for defeasible argumentation. In D. Gabbay and F. Guenthner (eds.),*Handbook of Philosophical Logic*, second edition, Vol 4, pp. 219-318. Kluwer Academic Publishers, Dordrecht etc., 2002.

[13] A. Wyner, K. Atkinson and T. Bench-Capon. Towards a structured online consultation tool. *Electronic Participation - Third International Conference (ePart (2011)*, volume 6847 of *Lecture Notes in Computer Science (LNCS)*, pages 286-297, Berlin, Germany, August 2011. Springer.