Chapter 5

CumulA:
a model of argumentation in stages

The previous chapters dealt with the nature of the rules and reasons that are at the basis of argumentation. In this chapter, we investigate the process of argumentation itself. We focus on arguments and their defeat. This leads to a formal model of argumentation in stages, called CumulA.[1]

In section 1, we introduce argumentation with defeasible arguments and give an overview of CumulA. In section 2, arguments and their structure are treated. In section 3, we discuss how the defeat of arguments is formalized using defeaters. In section 4, the stages of argumentation are characterized. Section 5 deals with lines of argumentation and argumentation diagrams. Section 6 gives a number of examples.

1 Argumentation in stages

Below, we first give an informal introduction of the key terminology, related to arguments and defeat as it is used in this chapter. Second, we give an overview of the formal model CumulA.

1.1 Arguments and defeat

The goal of argumentation is to find (rationally) *justified* conclusions (cf., e.g., Pollock, 1987). For instance, if a colleague enters the room completely soaked and tells that it is raining outside, I would of course conclude that it is wise to put on a raincoat. My conclusion is rationally justified, since I can give *support* for it, namely the fact that my colleague is completely soaked and tells me that it is raining. If I were asked why I concluded that it is wise to put on a raincoat, I could answer with the following *argument*:

---

[1] The name CumulA is an abbreviation of Cumulative Argumentation, but was chosen since it reminds of a certain type of cloud, the cumulus. The formal model is based on previous work (Verheij, 1995a, b, c). However, most definitions are new or have been changed considerably.

A colleague is completely soaked and tells that it is raining.
So, it is probably raining.
So, it is wise to put on a raincoat.

Such an argument is a reconstruction of how a conclusion can be supported. The argument given here consists of two *steps*. In general, an argument can support its conclusion if the steps in the argument are based on *rules* (see also chapter 4, section 1). Here we do not answer the question which argument steps can give rise to arguments that support their conclusion and which do not, or, in other words, which steps are based on rules and which are not. We assume that the rules allowing argument steps are somehow given.[2]

An argument that supports its conclusion does not always justify it. For instance, if in our example I look out the window and see wet streets, but otherwise a completely blue sky, I would conclude that the brief shower is over. So, while at the time my colleague entered it was justified for me to conclude that it is wise to put on a raincoat, it is not justified anymore after looking out the window. In this case, we say that the argument is *defeated*. In the example, the argument

A colleague is completely soaked and tells that it is raining.
So, it is probably raining.

does not justify its conclusion because of the argument

The streets are wet, but the sky is completely blue.
So, the shower is over.

In this case the argument that it is probably raining is defeated by the argument that the shower is over. The new information that the shower is over has the effect that the argument does not justify its conclusion, but does not change the fact that in principle the argument supports its conclusion.[3]

Our example has illustrated two points about argumentation, that form the basis of our model:

1. Argumentation is a process (see also chapter 1, section 1), in which at each stage new arguments are taken into account.

[2] We believe that in the end the rules and reasons on which argument steps are based are a special kind of *memes* (cf. Dawkins, 1989). An interesting account of the relation between rationality and evolution is given by Rescher (1988, p. 176ff.).

[3] It should be recalled that in our use of terminology justification of a conclusion does not imply truth of the conclusion (cf. chapter 1, section 1).

2. Each argument that is taken into account has either one of two statuses: the argument is either undefeated or defeated, indicating that the argument justifies its conclusion, or not, respectively.[4]

Our example showed that the status of an argument can depend on the structure of the argument, the counterarguments that are taken into account, and the argumentation stage.

- *The structure of the argument*
  The two-step argument that it is wise to put on a raincoat is defeated because already its first step, in which it is concluded that it is probably raining, is defeated.

- *Counterarguments*
  The argument that the shower is over defeats the argument that it is probably raining.

- *The argumentation stage*
  The argument that it is probably raining is only defeated once the argument that the shower is over has been taken into account.

1.2 Overview of CumulA

In this chapter, a formal model of argumentation with defeasible arguments is developed. This model is called CumulA. Formally, it builds on Lin and Shoham's Argument Systems (Lin and Shoham, 1989; Lin, 1993), Vreeswijk's Abstract Argumentation Systems (Vreeswijk, 1991, 1993), and Dung's Argumentation Frameworks (Dung, 1993, 1995). Key definitions in CumulA are those of *arguments*, *defeaters*, *argumentation theories*, *stages* and *lines of argumentation*.

- *Arguments*
  Arguments in CumulA are tree-like structures that represent how a conclusion is supported. Arguments are the subject of section 2. Our composite arguments are not, as usual, only constructed by subordination, but also by coordination. For instance, Lin and Shoham (Lin and Shoham, 1989; Lin, 1993) and Vreeswijk (1991, 1993) use subordination, but not coordination in their argumentation models. We investigate how the coordination of arguments is related to defeat.

[4] Cf. Pollock (1987-1995) and Vreeswijk (1991, 1993). Prakken (1993a, b) considers a third status: an argument can be defensible arguments, which means that it is neither undefeated nor defeated.

- *Defeaters*
  Defeaters indicate which arguments can defeat which other arguments. They consist of a set of challenging arguments and a set of challenged arguments. If the challenging arguments of a defeater are undefeated, they defeat its challenged arguments.[5] Defeaters are treated in section 3. We show that our defeaters can represent a wide range of types of defeat. Our defeaters are formally related to Dung's (1995) attacks. However, our defeaters can represent how the structure of arguments is related to defeat, and can represent more general types of defeat in which groups of arguments challenge other groups of arguments.

- *Argumentation theories*
  Argumentation depends on the language that is used, on the arguments that can support conclusions, and on which arguments defeat which other arguments. This information is represented as an argumentation theory. An argumentation theory consists of a set of sentences (together forming the language), a set of rules that give rise to arguments, and a set of defeater schemes that determine which arguments defeat which other arguments. In order to define forward and backward lines of argumentation (see below), an argumentation theory does not fix the premises of the arguments, as for instance in Lin and Shoham's (Lin and Shoham, 1989; Lin, 1993) and Vreeswijk's (1991, 1993) argumentation models. Argumentation theories are characterized at the end of section 3.

- *Stages*
  A stage in the argumentation process is characterized by the arguments that have been taken into account, and by the defeat status of these arguments, either undefeated or defeated. Which stages are allowed is determined by an argumentation theory. A stage consists of a pair of sets, one of them representing the arguments that are undefeated at the stage, the other the arguments that are defeated at the stage. The union of these sets represents which arguments have been taken into account. Stages are discussed in section 4. Vreeswijk's (1991, 1993) argument structures are comparable to our stages. However, they are representations of the arguments currently undefeated, and not of all arguments currently taken into account, whether undefeated or defeated.

- *Lines of argumentation*
  Argumentation can proceed in many ways, depending on the obtaining goals, protocols and strategies of argumentation. This leads to different lines of argumentation. A line of argumentation is a sequence of consecutive

5 Our notion of defeat is *counterargument-triggered*, as, e.g., Dung's (1995), and not *inconsistency-triggered*, as, e.g., Vreeswijk's (1991, 1993). We discuss this distinction more extensively in chapter 6, section 4.

argumentation stages. In a line of argumentation, arguments are gradually constructed. At each stage in a line of argumentation, the arguments taken into account have a defeat status. However, the status of an argument can change during a line of argumentation. Which lines of argumentation are possible is determined by an argumentation theory. We also define argumentation diagrams that represent several possible lines of argumentation as allowed by an argumentation theory. Lines of argumentation and argumentation diagrams are defined in section 5. Our lines of argumentation are related to Vreeswijk's (1991, 1993) argumentation sequences. However, since our argumentation theories do not fix the allowed premises, in our lines of argumentation arguments can be constructed not only forwardly, but also backwardly.

We have to make a disclaimer here: our definition of lines of argumentation does not prescribe how argumentation should proceed, but only attempts to describe which lines of argumentation are possible.[6] We return to this issue in section 5.3.

2 Arguments and their structure

This section deals with the structure of arguments. We treat arguments as tree-like structures of sentences, similar in form to logical proofs. After an informal discussion of elementary and composite argument structures (sections 2.1 and 2.2), we give a formal definition of arguments in section 2.3. The section ends with the definition of initials and narrowings of arguments (section 2.4).

2.1 Elementary argument structures

The simplest type of argument is the *statement*. Examples of statements are:

The sky is blue.

and

The film was good.

In principle, any (assertive) sentence can be used as a statement, so schematically statements have the following trivial structure:

Sentence.

[6] Recently several protocols prescribing (or at least constraining) lines of argumentation have been proposed, especially in a dialogical setting (see, e.g., Gordon, 1993a, 1993b, 1995; Brewka, 1994; Lodder and Herczog, 1995).

Some would hesitate to call statements arguments, because of their trivial structure. Since statements can be considered as the beginning of all argumentation, it will turn out convenient to include statements in the definition of arguments.

The simplest type of argument with non-trivial structure is the *single-step argument*, for instance:

The sun is shining.
So, it is a beautiful day.

Schematically, a single-step argument has the following structure:

Reason.
So, *Conclusion*.

A reason in an argument can consist of a several subreasons, as for instance in the following argument that has two subreasons:

Alex has an appointment at eight with John in Maastricht, John has an appointment at seven with Mary in Amsterdam.
So, John cannot keep both appointments.

Schematically, we have:

$Subreason_1$, $Subreason_2$, ..., $Subreason_n$.
So, *Conclusion*.

We use different terms 'subreason' and 'reason', since only the combination of the subreasons provides a reason that supports the conclusion. It should be noted that this is in contrast with everyday language, where the distinction between subreasons and reasons is not made, and both are called reasons.

2.2 Composite argument structures

Arguments can be combined. There are two basic ways to combine arguments into more complex structures, namely subordination and coordination.[7]

- Subordination of arguments

  If a single-step argument has a conclusion that is the same as one of the reasons or subreasons of another argument, arguments can be subordinated. We have already seen an example of subordination, namely:

---

[7] We use the same argument structure as Van Eemeren *et al*. (1981, 1987), but our terminology is different. Our coordinated arguments correspond to their multiple arguments.

A colleague is completely soaked and tells that it is raining.
So, it is probably raining.
So, it is wise to put on a raincoat.

This argument is the result from the subordination of the arguments

A colleague is completely soaked and tells that it is raining.
So, it is probably raining.

and

It is probably raining.
So, it is wise to put on a raincoat.

Schematically,

Reason.
So, $Conclusion_1$.
So, $Conclusion_2$.

- Coordination of arguments

If two arguments have the same conclusion, they can be coordinated. For instance, if the arguments

The sun is shining.
So, it is a beautiful day.

and

The sky is blue.
So, it is a beautiful day.

are coordinated, we obtain

The sun is shining; The sky is blue.
So, it is a beautiful day.

Schematically, we have

$Reason_1$; $Reason_2$.
So, *Conclusion*.

It should be noted that, in contrast with the subreasons mentioned earlier, each reason in a coordinated argument supports the conclusion on its own. To distinguish reasons and subreasons, reasons are separated by semicolons, while subreasons are separated by commas.[8] For instance, an argument can have the following structure:

$Subreason_{11}$, $Subreason_{12}$; $Subreason_{21}$, $Subreason_{22}$.
So, *Conclusion*.

Here $Subreason_{11}$ and $Subreason_{12}$ together form a reason for the conclusion, while $Subreason_{21}$ and $Subreason_{22}$ form a separate, second reason for it.

By repeating these two ways of combining arguments, the structure of arguments can become arbitrarily complex.

2.3 Definition of arguments

In our model of argumentation, we abstract from the language, and therefore treat a language simply as a set without any structure. Here we follow Lin and Shoham (Lin and Shoham, 1989; Lin, 1993), who use an unstructured language closed under negation, and Vreeswijk (1991, 1993), who uses an unstructured language containing a special sentence denoting contradiction.

Rules in a given language consist of a condition and a conclusion, which are formally a set of sentences and a sentence of the language, respectively. Since in our model all arguments are defeasible, we do not distinguish rules that give rise to strict arguments and rules that give rise to defeasible arguments.

Definition 1.

A *language* is any set, the elements of which are called the *sentences* of the language.[9] If *Subreasons* is a non-empty finite subset of a language *Language* and *Conclusion* is an element of *Language*, then

$Subconditions \rightarrow Conclusion$

is a *rule* of the language *Language*.[10] The set of rules of a language *Language* is denoted as Rules(*Language*).

Fixing a language *Language*, we obtain the following formal definition of arguments in the language. Our definition of arguments in a language is related to

[8] This convention is similar to the conventions in the logical programming language Prolog. In fact, a simple correspondence can be given between trees of Prolog clauses and our arguments.

[9] As elements of a set *Language*, sentences are just unspecified sets. We need one formal property to avoid ambiguity: there are no sentences $Sentence_0$, ..., and $Sentence_n$ (for some natural number n), such that $Sentence_0 \in \ldots \in Sentence_n$.

[10] Within set theory, a rule can be defined as an ordered pair (*Subreasons*, *Conclusion*).

the definitions of Lin and Shoham (Lin and Shoham, 1989; Lin, 1993) and Vreeswijk (1991, 1993), but does not presuppose a set of rules. Moreover, our definition allows not only the subordination, but also the coordination of arguments. Later (definition 4) we define the rules of an argument.

Definition 2.

The set of *arguments* in the language *Language* is the smallest set such that the following hold:

1. If *Sentence* is a sentence of the language *Language*, then
   Sentence
   is an argument. The conclusion of the argument *Sentence* is *Sentence*.
2. If *Conclusion* is an element of *Language* and $Argument_1, ..., Argument_n$ are arguments, then
   $\{\{Argument_1, ..., Argument_n\}\} \rightarrow Conclusion$
   is an argument. The conclusion of this argument is *Conclusion*.
3. If $\{Arguments_1\} \rightarrow Conclusion, ..., \{Arguments_n\} \rightarrow Conclusion$ are arguments, then
   $\{Arguments_1, ..., Arguments_n\} \rightarrow Conclusion$
   is an argument. The conclusion of this argument is *Conclusion*.

The conclusion of an argument *Argument* is denoted as Conclusion(*Argument*).

The first part of the recursive definition allows statements as arguments, the second allows subordination of arguments, and the third coordination. The previously discussed argument structures are all captured by this definition. An overview is given in Table 1. The abundance of brackets { } is required to distinguish reasons and subreasons: reasons are represented as sets, and coordinated reasons as sets of sets.[11]

It may seem that there is an ambiguity between a rule and a single-step argument. However, a rule has the form $SetOfSentences \rightarrow Sentence$, where *SetOfSentences* is a set of sentences, while a single-step argument has the form $SetOfSetsOfSentences \rightarrow Sentence$, where *SetOfSetsOfSentences* is a set of sets of sentences.[12]

[11] We use sets instead of sequences, as used by Lin and Shoham (Lin and Shoham, 1989; Lin, 1993) and Vreeswijk (1991, 1993), since changing the order of reasons or subreasons does not change an argument.

[12] Here we need the property mentioned in note 9.

Informal argument structure	Formal argument structure
Sentence.	*Sentence*
Reason. So, *Conclusion*.	$\{\{Reason\}\} \rightarrow Conclusion$
$Subreason_1$, ..., $Subreason_n$. So, *Conclusion*.	$\{\{Subreason_1, ..., Subreason_n\}\} \rightarrow Conclusion$
Reason. So, $Conclusion_1$. So, $Conclusion_2$.	$\{\{\{\{Reason\}\} \rightarrow Conclusion_1\}\} \rightarrow Conclusion_2$
$Reason_1$; $Reason_2$. So, *Conclusion*.	$\{\{Reason_1\}, \{Reason_2\}\} \rightarrow Conclusion$
$Subreason_{11}$, $Subreason_{12}$; $Subreason_{21}$, $Subreason_{22}$. So, *Conclusion*.	$\{\{Subreason_{11}, Subreason_{12}\},$ $\{Subreason_{21}, Subreason_{22}\}\}$ $\rightarrow Conclusion$
$Subreason_{11}$, ..., $Subreason_{1n_1}$; ...; ...; ...; $Subreason_{m1}$, ..., $Subreason_{mn_m}$. So, *Conclusion*.	$\{\{Subreason_{11}, ..., Subreason_{1n_1}\},$ ..., $\{Subreason_{m1}, ..., Subreason_{mn_m}\}\}$ $\rightarrow Conclusion$

Table 1: Overview of informal and formal argument structures

The following definitions of the premises and the rules of an argument follows the recursive structure of the definition of arguments. Vreeswijk's (1991, 1993) definitions are similar in style.

Definition 3.

If *Argument* is an argument, the set of *premises* of the argument, denoted as Premises(*Argument*), is defined recursively as follows:

1. Premises(*Sentence*) = {*Sentence*}, where *Sentence* is a sentence.
2. $\text{Premises}(\{\{Argument_1, ..., Argument_n\}\} \rightarrow Conclusion) =$
   $\text{Premises}(Argument_1) \cup ... \cup \text{Premises}(Argument_n)$,
   where $Argument_1$, ..., and $Argument_n$ are arguments (for some natural number n), and *Conclusion* is a sentence.

3. Premises({$Arguments_1$, ..., $Arguments_n$} → *Conclusion*) =
   Premises[$Arguments_1$] ∪ ... ∪ Premises[$Arguments_n$],[13]
   where $Arguments_1$, ..., and $Arguments_n$ are sets of arguments (for some natural number n), and *Conclusion* is a sentence.

Definition 4.

If *Argument* is an argument, the set of *rules* of the argument, denoted as Rules(*Argument*), is defined recursively, as follows:

1. Rules(*Sentence*) = ∅, where *Sentence* is a sentence.
2. Rules({{$Argument_1$, ..., $Argument_n$}} → *Conclusion*) =
   {{Conclusion($Argument_1$), ..., Conclusion($Argument_n$)} → *Conclusion*}
   ∪ Rules($Argument_1$) ∪ ... ∪ Rules($Argument_n$),
   where $Argument_1$, ..., and $Argument_n$ are arguments, and *Conclusion* is a sentence.
3. Rules({$Arguments_1$, ..., $Arguments_n$} → *Conclusion*) =
   Rules[$Arguments_1$] ∪ ... ∪ Rules[$Arguments_n$],
   where $Arguments_1$, ..., and $Arguments_n$ are sets of arguments, and *Conclusion* is a sentence.

Rules are not the same as single-step arguments: the single-step argument {{$Subreason_1$, ..., $Subreason_n$}} → *Conclusion* has one rule, namely {$Subreason_1$, ..., $Subreason_n$} → *Conclusion*.

Next we define argument schemes and their instances. Argument schemes are basically arguments that can contain wildcards. An instance of an argument scheme is obtained by 'filling in' each occurrence of the wildcard ∗.

Argument schemes are useful to denote arguments that have a common part, such as the same final step. For instance, all arguments with an equal final step, informally denoted as

...
So, *Reason*.
So, *Conclusion*.

are represented by the argument scheme {{∗*Reason*}} → *Conclusion*.

In the definition of argument schemes, the wildcard ∗ has different roles depending on its position in the argument scheme. The argument scheme ∗*Conclusion* represents an argument with conclusion *Conclusion*. Some of the instances of ∗*Conclusion* are *Conclusion* and {{*Reason*}} → *Conclusion*. In {{∗}} → *Conclusion*, the wildcard represents any argument. Some instances are {{*Reason*}} → *Conclusion* and {{{{$Reason_1$}} → $Reason_2$}} → *Conclusion*. In {∗} → *Conclusion*,

[13] For a function F and a set *Set* that is a subset of the domain of F, F[*Set*] denotes the image of *Set* under F.

the wildcard represents any (finite) set of arguments. Some instances are $\{\{Reason_1\}\} \rightarrow Conclusion$ and $\{\{Reason_1\}, \{Reason_2\}\} \rightarrow Conclusion$.

Formally, argument schemes and their instances scheme are defined as follows.

Definition 5.

The set of *argument schemes* in the language *Language* is the smallest set such that the following hold:

1. If *Sentence* is an element of *Language*, then *Sentence*, $*Sentence$, and $*$ are argument schemes.
2. If $ArgumentScheme_1, ..., ArgumentScheme_n$ are argument schemes, then $\{\{ArgumentScheme_1, ..., ArgumentScheme_n\}\} \rightarrow Conclusion$ and $\{*\} \rightarrow Conclusion$ are argument schemes.
3. If $\{ArgumentSchemes_1\} \rightarrow Conclusion, ..., \{ArgumentSchemes_n\} \rightarrow Conclusion$ are argument schemes, then $\{ArgumentSchemes_1, ..., ArgumentSchemes_n\} \rightarrow Conclusion$ is an argument scheme.

Definition 6.

The *instances* of an argument scheme *ArgumentScheme* in the language *Language*, denoted as Instances(*ArgumentScheme*), are defined recursively, as follows:[14]

1. Instances(*Sentence*) = {*Sentence*}
   Instances($*Sentence$) =
   {*Argument* | *Argument* is an argument of *Language* with conclusion *Sentence*}
   Instances($*$) = {*Argument* | *Argument* is an argument of *Language*}
2. $\text{Instances}(\{\{ArgScheme_1, ..., ArgScheme_n\}\} \rightarrow Conclusion) =$
   $\text{Instances}(ArgScheme_1) \cup ... \cup \text{Instances}(ArgScheme_n)$
   $\text{Instances}(\{*\} \rightarrow Conclusion) =$
   $\{Arguments \rightarrow Conclusion$ | *Arguments* is a set of arguments of *Language*}
3. $\text{Instances}(\{ArgSchemes_1, ..., ArgSchemes_n\} \rightarrow Conclusion) =$
   $\text{Instances}[ArgSchemes_1] \cup ... \cup \text{Instances}[ArgSchemes_n]$

Any argument is an argument scheme, whose only instance is the argument itself.

2.4 Initials and narrowings of arguments

In this section, we discuss the initials and the narrowings of an argument. They are purely determined by the structure of the argument.

Arguments can have other arguments as initial parts. For instance, the argument

[14] This definition is recursive in the structure of arguments, just as the definitions of premises, rules and argument schemes. For brevity, we do not explicitly state that *Sentence* is a sentence, that $Argument_i$ is an argument (for i = 1, ..., n) etc.

A colleague is completely soaked and tells that it is raining.
So, it is probably raining.
So, it is wise to put on a raincoat.

has the argument

A colleague is completely soaked and tells that it is raining.
So, it is probably raining.

as an initial part. Formally the initials of an argument are defined as follows.

Definition 7.

If *Argument* is an argument, the set of *initials* of the argument, denoted as Initials(*Argument*), is defined recursively as follows

1. Initials(*Sentence*) = $\varnothing$
2. Initials({{$Argument_1$, ..., $Argument_n$}} $\to$ *Conclusion*) = Initials($Argument_1$) $\cup$ ... $\cup$ Initials($Argument_n$) $\cup$ {$Argument_1$, ..., $Argument_n$}
3. Initials({$Arguments_1$, ..., $Arguments_n$} $\to$ *Conclusion*) = Initials({$Arguments_1$} $\to$ *Conclusion*) $\cup$ ... $\cup$ Initials({$Arguments_n$} $\to$ *Conclusion*)

The initials of an argument are also arguments. The definition shows that an argument is not an initial of itself and that all arguments, except for statements, have initials.

If the conclusion of an argument is supported by a coordinate argument with separate reasons, one or more of the reasons can be removed from the argument. For instance, if the reason ‘The sun is shining’ is removed from the argument

The sun is shining; The sky is blue.
So, it is a beautiful day.

we obtain the argument

The sky is blue.
So, it is a beautiful day.

The latter argument is called a narrowing of the former. Only if one allows the coordination of arguments, it is possible to define the narrowings of an argument. Formally, the narrowings of an argument are defined as follows.

Definition 8.

If *Argument* is an argument, the set of *narrowings* of the argument, denoted as Narrowings(*Argument*), is defined recursively as follows:

1. Narrowings(*Sentence*) = $\emptyset$
2. Narrowings({{$Argument_1$, ..., $Argument_n$}} $\rightarrow$ *Conclusion*) =
   {{{$Narrowing_1$, ..., $Narrowing_n$}} $\rightarrow$ *Conclusion* |
   $Narrowing_i \in$ Narrowings($Argument_i$) for all i = 1, ..., n}
3. Narrowings({$Arguments_1$, ..., $Arguments_n$} $\rightarrow$ *Conclusion*) =
   {$W \rightarrow$ *Conclusion* | $\emptyset \subset W \subset$ {$Arguments_1$, ..., $Arguments_n$}}[15]

If $Argument_1$ is a narrowing of $Argument_2$, then $Argument_2$ is a *broadening* of $Argument_1$.

The definition shows that in a narrowing of an argument the final conclusion is supported by less reasons than in the argument itself (part 3 of the definition). In a narrowing of an argument, also the intermediate conclusions can be supported by less reasons (part 2 of the definition).

The narrowings of arguments are also arguments. If follows from the definition that arguments are not narrowings of themselves, and that not all arguments have narrowings. The conclusion of a narrowing of an argument is equal to the conclusion of the argument. As a result, no narrowing of an argument is at the same time an initial of the argument.

3 Defeat and defeaters

In the previous section, we saw that arguments are in form comparable to proofs. However, there is a major difference between arguments and proofs. While proofs justify their conclusions under all circumstances, arguments do not: arguments can be defeated.

In this section, we deal with the defeat of arguments. We distinguish several types of defeat and corresponding defeaters. These indicate which arguments can defeat which other arguments (sections 3.1 to 3.5). Then we discuss the role of defeater schemes (section 3.6). This leads to the formal definition of defeaters and defeater schemes (section 3.7).

3.1 Undercutting defeat

The first type of defeat that we discuss is defeat by an undercutter.[16] As an example, we consider the following (single-step) argument:

The object looks red.
So, the object is red.

[15] $V \subset W$ means that V is a proper subset of W.
[16] Pollock (1987-1995) has argued for the distinction between defeat by an undercutter and by a rebutter (discussed in section 3.2).

In principle this argument supports its conclusion, but suppose that we also have the following argument (a statement):

The object is illuminated by a red light.

Taking both arguments into account, the argument that the object is red does no longer justify its conclusion. Because the object is illuminated by a red light, the fact that it looks red is no longer a reason for the conclusion that the object is red. (Of course the object can still be red, but we cannot justify this by the fact that it looks red.) We say that the fact that the object is illuminated by a red light *undercuts* the argument that the object is red.[17]

In our formal model, this fact is represented as follows:

Illuminated_by_a_red_light [{{Looks_red}} → Is_red]

This is an example of a *defeater*, the formal definition of which follows later.[18] Informally, the defeater represents that if the argument on the left, Illuminated_by_a_red_light, is undefeated, it defeats the argument on the right {{Looks_red}} → Is_red.[19] To emphasize that the latter argument becomes defeated, it is put between square brackets [ ].

3.2 Rebutting defeat

The second type of defeat is rebuttal.[20] For instance, if John likes French fries, but is on a low calorie diet, we have the following two arguments:

John likes French fries.
So, he orders French fries.

and

[17] The example has been used at several occasions by Pollock as an illustration of undercutting defeat (e.g., Pollock, 1986, p. 39ff.; 1994).

[18] In our terminology, a defeater is not itself an argument or a reason that challenges another argument (as for instance Pollock uses the term), but a relation between challenging and challenged arguments.

[19] Note that the arguments Illuminated_by_a_red_light and {{Looks_red}} → Is_red do not have inconsistent conclusions. The example shows an important choice underlying the CumulA model: the defeat of arguments is in CumulA not *inconsistency-triggered*, but *counterargument-triggered* (see note 5). Not inconsistency, but counterargument (represented by defeaters) is the primitive notion in CumulA. We come back to this distinction in chapter 6, section 4.

[20] See note 16.

John is on a low calorie diet.
So, he does not order French fries.

Assuming that people who are on a diet try to suppress their eating impulses, John probably does not order fries, since the latter argument would be more important. In this case, the former argument is defeated by the latter. Formally, this would be represented by the following defeater:

{{On_low_calorie_diet}} → Not_order_fries [{{Likes_fries}} → Order_fries]

The argument on the left, {{On_low_calorie_diet}} → Not_order_fries, defeats the argument in square brackets on the right, {{Likes_fries}} → Order_fries. If, as in this example, an argument defeats an argument with opposite conclusion, we speak of *rebutting* defeat.

3.3 Defeat by sequential weakening

The third type of defeat is defeat by sequential weakening. An example of this is the following argument, based on the well-known sorites paradox:[21]

This body of grains of sand is a heap.
So, this body of grains of sand minus 1 grain is a heap.
So, this body of grains of sand minus 2 grains is a heap.
...
So, this body of grains of sand minus *n* grains is a heap.

Each single step of the argument is correct, but clearly the argument cannot be pursued indefinitely, since in the end there is no grain of sand left. For *n* large enough, the argument above does clearly not justify its conclusion and should be defeated. The important point here is that it is impossible to choose a single step that makes the argument defeated. Only because the step is repeated too often, the argument is weakened below the limit of acceptability, and is defeated.

Since argument steps normally can be chained, we need a way to represent the fact that certain sequences of steps can lead to the defeat of an argument. A defeater representing the situation of our example has the following form:

[Body_of_sand_is_heap → Body_of_sand_minus_1_grain_is_heap
→ Body_of_sand_minus_2_grains_is_heap
→ ...
→ Body_of_sand_minus_*n*_grains_is_heap]

For convenience, we have left out the brackets { }.

[21] Read (1995, p. 173ff.) discusses philosophical issues related to the sorites paradox.

In the example, there is an argument that is clearly defeated because it contains an unacceptable sequence of steps, but that does not contain one single argument step that is to blame. In such a case, we speak of defeat by *sequential weakening* of the argument.[22]

3.4 Defeat by parallel strengthening

The fourth type of defeat is defeat by parallel strengthening. Assume that John has committed an offense, but is a minor first offender. As a result, the judge might consider the following argument:

John is a minor first offender.
So, John should not be punished.

If for instance John has robbed Alex, the judge might consider this an argument that rebuts the following argument with opposite conclusion:

John has robbed Alex.
So, John should be punished.

In the case of rebuttal the judge decided not to punish John. The judge might decide analogously if John had injured Alex in a fight.

However, if John has both robbed Alex and injured him in a fight, the judge might decide differently. Since there are now two reasons for punishing John, coordination of the arguments gives us the following composite argument:

John has robbed Alex; John injured Alex in a fight.
So, John should be punished.

This argument might defeat the argument that John should not be punished. In that case, the argument that John should be punished defeats another argument, while its narrowings do not. A defeater representing this is the following:

{{Robbed}, {Injured}} → Punished [{{Minor_first_offender}} → Not_punished]

The argument {{Robbed}, {Injured}} → Punished defeats the argument {{Minor_first_offender}} → Not_punished.[23] In this example the defeat of the argument not to punish John can be explained by the *parallel strengthening* of the argument to punish him. We speak of defeat by parallel strengthening if an

[22] The term is taken from Verheij (1995c).
[23] In Reason-Based Logic, this example would involve the weighing of reasons (chapter 2, sections 1.3 and 3.3). Cf. Verheij (1994).

argument that has narrowings defeats another argument, while the narrowings themselves do not.[24]

3.5 Collective and indeterministic defeat

All examples of defeaters that we have seen consisted of a single challenging and a single challenged argument. Such defeaters are called simple. However, there are cases in which groups of arguments must be considered. We discuss two types of situations in which this is the case, namely *collective defeat* and *indeterministic defeat*. It turns out that in order to represent these types of defeat, we need compound defeaters, that consist of groups of challenging and challenged arguments.

Collective and indeterministic defeat occur if there is a number of arguments that can clearly not all justify their conclusions, for instance because their conclusions cannot all hold, but neither of which is clearly defeated by any of the others. We give an example.

It can be the case that an employer wants to hire two persons if they are qualified. If John is qualified, the employer can make the following argument:

John is qualified. So, John is hired.

On its own, this argument can be undefeated, but now assume that not only John, but also Alex and Mary are qualified for the job. As a result, the employer can also make the following two arguments:

Alex is qualified. So, Alex is hired.
Mary is qualified. So, Mary is hired.

Since the employer only wants to hire two persons, the three arguments cannot all be undefeated.

If there is no additional information to resolve this conflict of arguments, two approaches can be distinguished that nevertheless 'magically' resolve the conflict: collective and indeterministic defeat.

In the first approach to dealing with the unresolved conflict of arguments, collective defeat,[25] all arguments are considered defeated. We speak of collective defeat if a group of arguments is defeated as a whole, while the arguments in the

[24] The example given here is also an example of rebutting defeat, showing that the discussed types of defeat can overlap.

[25] The term 'collective defeat' stems from Pollock (1987). Our collective defeat generalizes his. Pollock's collective defeat is a general principle to preserve consistency. It makes groups of otherwise undefeated, but conflicting arguments defeated. Our collective defeat is optional, depending on the compound defeaters of a particular argumentation theory, and can occur for any group of arguments, not only conflicting.

group would on their own not be defeated. A defeater representing the situation in our example could have the following form:

[{{John_is_qualified}} → John_is_hired,
{{Alex_is_qualified}} → Alex_is_hired,
{{Mary_is_qualified}} → Mary_is_hired]

In this defeater, all arguments are inside the square brackets indicating that they are defeated as a group. We say that this defeater is *right-compound*, since it has more than one challenged argument.

A defeater such as the one above represents that the arguments inside the square brackets are defeated as a group, and not simply that they are all three defeated. The latter would be represented by the following three simple defeaters:

[{{John_is_qualified}} → John_is_hired]
[{{Alex_is_qualified}} → Alex_is_hired]
[{{Mary_is_qualified}} → Mary_is_hired]

The difference with the compound defeater above is that the compound defeater only represents that the group of three should be defeated if otherwise neither of the arguments in the group would be defeated. If the argument that Mary is hired for the job is defeated *for another reason* (i.e., because of another defeater), for instance, that she prefers a job somewhere else, the compound defeater above does not anymore imply the defeat of the argument that John is hired for the job. Only if all three arguments would otherwise be undefeated, the compound defeater results in their defeat as a group. The three simple defeaters would not have the same effect: they represent that the arguments are defeated anyway.

The second approach to dealing with the unresolved conflict of arguments is indeterministic defeat. In this approach, the conflict is resolved by considering one of the arguments in the conflict defeated. Since there are several choices that can be made, neither of which is better than the others, the conflict is 'indeterministically' solved: each choice of a defeated argument is allowed. In the example, there are three solutions, represented by the following defeaters:

{{John_is_qualified}} → John_is_hired, {{Alex_is_qualified}} → Alex_is_hired
[{{Mary_is_qualified}} → Mary_is_hired]
{{Alex_is_qualified}} → Alex_is_hired, {{John_is_qualified}} → John_is_hired
[{{Mary_is_qualified}} → Mary_is_hired]
{{Mary_is_qualified}} → Mary_is_hired, {{John_is_qualified}} → John_is_hired
[{{Alex_is_qualified}} → Alex_is_hired]

Each defeater represents that two of the arguments challenge the third, and can result in its defeat if they are both undefeated. We say that this defeater is *left-compound*, since it has more than one challenging arguments.

3.6 Defeater schemes

We have encountered several examples of defeaters. They all contained representations of full arguments. As we will see, this is not always convenient. As an example, we reconsider the argument that an object is red because it looks red. This argument was defeated by the statement that the object is illuminated by a red light. We had the following defeater:

Illuminated_by_a_red_light [{{Looks_red}} → Is_red]

But it can of course also be the case that the fact ‘The object is illuminated by a red light’ is not merely put forward as a statement, but is itself supported by some non-trivial argument, for instance as follows:

Ralph says that the object is illuminated by a red light.
So, the object is illuminated by a red light.

If this argument is not defeated, it defeats the argument that the object is red, just like the statement ‘The object is illuminated by a red light’ did. It does not matter how the conclusion that the object is illuminated by a red light is justified. By whatever argument that conclusion is justified, it defeats the argument that the object looks red.

Similarly, it can be the case that the argument step that the object is red because it looks red is itself part of a larger argument, for instance as follows:

The object reflects light of a particular wave length.
So, the object looks red.
So, the object is red.

This argument is defeated too if the conclusion that the object is illuminated by a red light is justified. (It should be noted that this does not imply that the argument

The object reflects light of a particular wave length.
So, the object looks red.

is defeated. The conclusion that the object looks red is still justified.)

This leads to the notion of *defeater schemes*. We want to represent that any argument that justifies the conclusion that the object is illuminated by a red light, defeats any argument that ends with the argument step that the object is red because it looks red. A defeater scheme representing this looks as follows:

∗Illuminated_by_a_red_light [{{∗Looks_red}} → Is_red]

Instead of arguments, a defeater scheme contains argument schemes, such as ∗Illuminated_by_a_red_light and {{∗Looks_red}} → Is_red. The defeater above will have the effect that any argument that is an instance of ∗Illuminated_by_a_red_light challenges any instance of {{∗Looks_red}} → Is_red.

In our example, this is just as required, since the argument scheme {{∗Looks_red}} → Is_red has both

{{Looks_red}} → Is_red

and

{{{{Reflect_light_of_particular_wave_length}} → Looks_red}} → Is_red

as instances.

3.7 Definition of defeaters and argumentation theories

Having finished the description of different types of defeat, we come to the formal definition of defeaters. We have seen several examples, all captured by the following definition.

Definition 9.

If $ChallengingArgument_1$, ..., $ChallengingArgument_n$, $ChallengedArgument_1$, ..., $ChallengedArgument_m$ are arguments, then

$ChallengingArgument_1$, ..., $ChallengingArgument_n$ [$ChallengedArgument_1$, ..., $ChallengedArgument_m$]

is a *defeater*. The arguments $ChallengingArgument_1$, ..., $ChallengingArgument_n$ are the *challenging arguments* of the defeater, the arguments $ChallengedArgument_1$, ..., $ChallengedArgument_m$ the *challenged arguments* of the defeater. A defeater with at most one challenging and at most one challenged argument is *simple*, otherwise *compound*. A defeater that has more than one challenging argument is *left-compound*, a defeater with more than one challenged argument *right-compound*.

Intuitively, the defeater $ChallengingArgument_1$, ..., $ChallengingArgument_n$ [$ChallengedArgument_1$, ..., $ChallengedArgument_m$] represents the fact that the arguments $ChallengingArgument_1$, ..., and $ChallengingArgument_n$ defeat the arguments $ChallengedArgument_1$, ..., $ChallengedArgument_m$, if they are themselves not defeated.

Our defeaters are related to Dung's (1993, 1995) attacks. However, our defeaters take the structure of arguments into account, and do not consist of a single challenging and a single challenged argument, but of a group of challenging and a group of challenged arguments.

We already discussed the need for defeater schemes. They contain argument schemes instead of arguments. Defeater schemes and their instances are formally defined as follows.

Definition 10.

If $\mathit{ChallengingArgScheme}_1$, ..., $\mathit{ChallengingArgScheme}_n$, $\mathit{ChallengedArgScheme}_1$, ..., $\mathit{ChallengedArgScheme}_m$ are argument schemes, then

$\mathit{ChallengingArgScheme}_1, \ldots, \mathit{ChallengingArgScheme}_n$
$[\mathit{ChallengedArgScheme}_1, \ldots, \mathit{ChallengedArgScheme}_m]$

is a *defeater scheme*. If each argument scheme in the defeater scheme is replaced by one of its instances, the resulting defeater is an *instance* of the defeater scheme.

Just as arguments are a special kind of argument schemes, defeaters are a special kind of defeater schemes, with as only instance the defeater itself.

We have described several types of defeat. In Table 2, we give an overview of these types of defeat and their corresponding defeater schemes.

Type of defeat	**Corresponding defeater scheme(s)**
Undercutting defeat	$*\mathit{Conclusion}_1\ [*\mathit{Reason} \rightarrow \mathit{Conclusion}_2]$
Rebutting defeat	$*\mathit{Pro} \rightarrow \mathit{Conclusion}\ [*\mathit{Con} \rightarrow \mathit{Not\_conclusion}]$
Defeat by sequential weakening	$[*\mathit{Sentence}_1 \rightarrow \ldots \rightarrow \mathit{Sentence}_n]$
Defeat by parallel strengthening	$\{\{*\mathit{Reason}_1\}, \ldots, \{*\mathit{Reason}_n\}\} \rightarrow \mathit{Conclusion}_1\ [*\mathit{Conclusion}_2]$
Collective defeat	$[*\mathit{Conclusion}_1, \ldots, *\mathit{Conclusion}_n]$
Indeterministic defeat	$*\mathit{Conclusion}_1\ [*\mathit{Conclusion}_2, \ldots, *\mathit{Conclusion}_n]$, ..., $*\mathit{Conclusion}_n\ [*\mathit{Conclusion}_1, \ldots, *\mathit{Conclusion}_{n-1}]$,

Table 2: Types of defeat and their corresponding defeater schemes

The types of defeat in this table are not disjoint, in the sense that there can be defeaters that are instances of defeater schemes of different types.

Argumentation depends on which arguments can support conclusions and on the situations in which arguments defeat other arguments. This is specified by an *argumentation theory*. A natural way to specify the arguments is by the rules from which they are constructed. A natural way to specify defeat situations is by defeater schemes. This gives us the following definition.

Definition 11.
An *argumentation theory* is a triple (*Language*, *Rules*, *DefeaterSchemes*), such that *Language* is a language, *Rules* is a set of rules of the language *Language*, and *DefeaterSchemes* is a set of defeater schemes of the language. Any argument that has only rules in *Rules* is an *argument* of the argumentation theory. Any instance of a defeater scheme in *DefeaterSchemes* is a *defeater* of the argumentation theory.

Our argumentation theories correspond to Lin and Shoham's argument structures (Lin and Shoham, 1989; Lin, 1993), Vreeswijk's (1991, 1993) abstract argumentation systems, and Dung's (1993, 1995) argumentation frameworks. Lin and Shoham and Vreeswijk include a set of premises. In CumulA, however, an argumentation theory does not specify premises, since in CumulA's lines of argumentation the premises can change (see section 5). Dung's definitions do not specify a set of premises either, but for the reason that they fully abstract from the structure of arguments.

4 Stages of the argumentation process

In the previous two sections, we discussed arguments and defeaters. They play a central role in argumentation: arguments represent how conclusions can be supported and defeaters represent which arguments can defeat other arguments. In this section, we discuss how defeaters determine the status of the arguments taken into account, i.e. which of them are defeated and which are undefeated. In the next subsection we treat how the status of an argument relates to the status of its initials and narrowings (section 4.1). Then we describe some notions that characterize the effects of a defeater (section 4.2). Thereafter we characterize the status of arguments in an argumentation stage (section 4.3). In section 4.4, the stages of an argumentation theory are formally defined.

4.1 Initials, narrowings and defeat

In this section, we encounter three general requirements that must hold for any defeat status assignment of the arguments that are taken into account at a stage of the argumentation process.

Every argument that is taken into account has one of two statuses: it can be either undefeated or defeated. By definition, an undefeated argument justifies its conclusion, while a defeated argument does not. So, the first requirement is simply that no argument can be defeated and undefeated at the same time. Obviously, an argument cannot both justify and not justify its conclusion.

The second requirement relates the statuses of an argument and its initials: if an initial of an argument is defeated, the argument is itself defeated. For instance, if the argument

The object looks red.
So, the object is red.

is defeated and does not justify the conclusion that the object is red, the argument

The object looks red.
So, the object is red.
So, the object attracts the attention.

is also defeated and cannot justify the conclusion that the object attracts the attention. (On the other hand, it is possible that the latter argument is defeated, while the former is not.) Generally, an argument can not justify its conclusion if an intermediate conclusion is not justified. In other words, an argument never withstands defeat better than its initials.

The third requirement relates the statuses of an argument and its narrowings: if a narrowing of an argument is undefeated, the argument is itself undefeated. For instance, if the argument

The sun is shining.
So, it is a beautiful day.

is undefeated and justifies its conclusion that it is a beautiful day, the argument

The sun is shining; The sky is blue.
So, it is a beautiful day.

cannot be defeated and not justify that same conclusion. Intuitively, an argument does not withstand defeat worse than an argument containing less reasons.[26] Only one of the narrowings of an argument needs to be undefeated in order to make it undefeated. For instance, the argument

The sky is blue.
So, it is a beautiful day.

---

[26] Pollock (1991) has argued against this so-called accrual of reasons. In chapter 6, section 2, we give our reply.

can be defeated (by some other argument), while the two arguments above are undefeated. Adding a reason can never make an argument defeated, but sometimes can make an argument undefeated, as in a case of defeat by parallel strengthening.

Summarizing we have the following three requirements for any defeat status assignment:

1. Each argument is either undefeated or defeated.
2. If an initial of an argument is defeated, the argument is defeated.
3. If a narrowing of an argument is undefeated, the argument is undefeated.

4.2 Relevant, triggered, respected and inactive defeaters

Defeaters play a central role in the determination of the defeat status of arguments. By default, an argument is undefeated, but defeaters can change this default status. Recall that defeaters indicate when undefeated arguments can defeat other arguments. We discuss four notions that are important for the effects of defeaters: a defeater can be *relevant*, *triggered*, *respected* and *inactive*.

As an example, we take the defeater

Ralph's_testimony → Illuminated_by_a_red_light [Looks_red → Is_red]

A defeater only can have effects if all arguments in it have been taken into account. If only the argument Looks_red → Is_red has been taken into account, the defeater above has no effect. Only if the argument Ralph's_testimony → Illuminated_by_a_red_light is also taken into account, can the argument Looks_red → Is_red be challenged. If all arguments in a defeater have been taken into account, the defeater is *relevant*.

A relevant defeater can only lead to the defeat of its challenged arguments if the challenging arguments are undefeated. Returning to our example, it can turn out that Ralph is lying, with the result that the argument Ralph's_testimony → Illuminated_by_a_red_light does not justify its conclusion, and is defeated (on the basis of some other defeater). Even though the argument that the object is illuminated by a red light is taken into account, it does not challenge the argument that the object is red, since it is itself defeated. If all challenging arguments of a relevant defeater are undefeated, the defeater is *triggered*.

Normally, if a relevant defeater is triggered, the challenged arguments are defeated. In our example, if the argument Ralph's_testimony → Illuminated_by_a_red_light is undefeated, the argument Looks_red → Is_red is defeated. In general, if all challenged arguments of a triggered defeater are defeated, the defeater is *respected*.

There is one situation in which a defeater of which the challenging arguments are undefeated do not lead to the defeat of its challenged arguments. This can only happen in a case of collective defeat (see section 3.5), represented by a right-compound defeater, when the challenged arguments are not defeated as a group, because some of them (but not all) are challenged arguments of another respected defeater.

As an example, we take the following two defeaters:

[John_is_qualified → John_is_hired, Mary_is_qualified → Mary_is_hired]
Mary_prefers_another_job [Mary_is_qualified → Mary_is_hired]

Here Mary_prefers_another_job represents that Mary prefers another job than the one for which both John and Mary are qualified. If now the two arguments

John_is_qualified → John_is_hired
Mary_is_qualified → Mary_is_hired

are taken into account, only one of the defeaters is relevant (the first), and should result in the defeat of both arguments. But if also the statement

Mary_prefers_another_job

is taken into account, the situation changes. Both defeaters are relevant. Since the argument Mary_prefers_another_job is not even challenged in one of the defeaters, it is undefeated and defeats the argument Mary_is_qualified → Mary_is_hired. But now the other defeater, that represents collective defeat, should not lead to the defeat of John_is_qualified → John_is_hired, since one of its challenged arguments is already defeated by another defeater. In this situation, we say that the defeater is *inactive*, otherwise *active*. Only active defeaters can lead to the defeat of their challenged arguments.

4.3 Stages and defeat

The stages of the argumentation process are characterized by the arguments that have been taken into account, and by the status the arguments have. The status of the arguments taken into account is determined by the defeaters of the argumentation theory. In this section, we discuss how.

An argument can be defeated in two ways: directly and indirectly. An argument is *directly defeated* if it is a challenged argument of a triggered active defeater. As an example, we consider the red light example again. We have the simple defeater

Illuminated_by_a_red_light [Looks_red → Is_red]

Assume that two arguments have been taken into account:

Illuminated_by_a_red_light
Looks_red → Is_red

Clearly, the defeater is relevant. The argument Illuminated_by_a_red_light cannot be defeated, since there is no defeater in which it is challenged. As a result, the defeater is triggered, and also active, since it is not left-compound. As a result, the argument Looks_red → Is_red is directly defeated by the argument Illuminated_by_a_red_light.

An argument is *indirectly defeated* if it has a defeated initial or broadening. Indirect defeat corresponds to the requirements on the statuses of arguments and their initials and narrowings (section 4.1). For instance, if in the example above the argument

Looks_red → Is_red → Attracts_attention

were also taken into account, it would be defeated, because its initial Looks_red → Is_red would have been defeated.

An argument can be both directly and indirectly defeated. For instance, if the argument

John_is_color_blind

were taken into account, and the theory contained the defeater scheme

John_is_color_blind [∗Is_red → Attracts_attention],

the argument Looks_red → Is_red → Attracts_attention would be both directly defeated, by the argument John_is_color_blind, and indirectly, by the (direct) defeat of its initial Looks_red → Is_red.

We have now discussed all ingredients of our definition of an argumentation stage. A stage is characterized by the arguments that are taken into account and by their statuses. As a result, a stage is defined as a defeat status assignment, that must obey the requirements of section 4.1. Furthermore, which arguments have the status undefeated and which defeated is determined by the defeaters, as follows:

> An argument has the status ‘defeated’ if and only if the argument is directly or indirectly defeated.

4.4 Definition of stages

We have seen that the status of initials and narrowings of an argument can have an effect on the status of the argument itself. The *range* of a set includes all arguments that can have such effects for the arguments of the set. Formally, the range of a set of arguments is defined as follows.

Definition 12.

The *range* of a set of arguments *Arguments*, denoted as Range(*Arguments*), is the smallest set of arguments *Range*, such that the following hold:

1. *Arguments* is a subset of *Range*.
2. Any initial of an argument in *Range* is an element of *Range*.
3. Any narrowing of an argument in *Range* is an element of *Range*.

A set of arguments that is equal to its range is a *range* of arguments.

We have discussed three requirements that are the result of the relations of the status of an argument and the statuses of its initials and narrowings. These requirements lead to the following definition of a *defeat status assignment*.

Definition 13.

A *defeat status assignment* of a range of arguments *Range* has the form

UndefeatedArguments (*DefeatedArguments*),

such that the following hold:

1. The arguments in *Range* are precisely the arguments in *UndefeatedArguments* and *DefeatedArguments*, but no argument is both in *UndefeatedArguments* and in *DefeatedArguments*.
2. No initial of an argument in *UndefeatedArguments* is an element of *DefeatedArguments*.
3. No narrowing of an argument in *DefeatedArguments* is an element of *UndefeatedArguments*.

The set *Range*, equal to the union of *UndefeatedArguments* and *DefeatedArguments*, is the *range* of the defeat status assignment.

Notation: A defeat status assignment of a finite range of arguments will often be denoted as

$UndefeatedArgument_1, \ldots, UndefeatedArgument_n$
$(DefeatedArgument_1, \ldots, DefeatedArgument_m)$

Our defeat status assignments are formally related to Pollock's (1994, 1995) partial status assignments, but have a different use. Pollock uses status assignments to be able to deal with certain problem cases. We use status assignments since they enable the definition of argumentation stages.

The second requirement in the definition of defeat status assignments is well-known and has a counterpart (in different forms) in many argumentation models that take the subordination of arguments into account, such as the models of Lin and Shoham (Lin and Shoham, 1989; Lin, 1993) and Vreeswijk (1991, 1993). The third requirement is, as far as we know, new in CumulA since in other models the coordination of arguments is not taken into account. It represents how the coordination of arguments is related to defeat.

Next we define when a defeater is relevant, triggered, respected and (in)active.

Definition 14.

A defeater *ChallengingArguments* [*ChallengedArguments*] is *relevant* for a defeat status assignment *UndefeatedArguments* (*DefeatedArguments*) if all arguments in *ChallengingArguments* and *ChallengedArguments* are in the range of the defeat status assignment.

Definition 15.

A defeater *ChallengingArguments* [*ChallengedArguments*] is *triggered* in the defeat status assignment *UndefeatedArguments* (*DefeatedArguments*) if it is relevant and *ChallengingArguments* is a subset of the range of *UndefeatedArguments*.

Definition 16.

A defeater *ChallengingArguments* [*ChallengedArguments*] is *respected* in the defeat status assignment *UndefeatedArguments* (*DefeatedArguments*) if *ChallengingArguments* is a subset of the range of *UndefeatedArguments* and *ChallengedArguments* is a subset of the range of *DefeatedArguments*.

Definition 17.

A defeater *ChallengingArguments* [*ChallengedArguments*] is *inactive* in the defeat status assignment *UndefeatedArguments* (*DefeatedArguments*) if it is relevant and there is a respected defeater *ChallengingArguments*' [*ChallengedArguments*'], such that some, but not all, arguments in *ChallengedArguments* are an element of, or have an initial or broadening in *ChallengedArguments*'. A relevant defeater is *active* if it is not inactive.

As immediate consequences of these definitions, triggered defeaters are always relevant, and respected defeaters are always triggered (and therefore relevant).

The following definition captures the direct and indirect defeat of arguments.

Definition 18.

The argument *Argument* is *defeated* by the arguments *ChallengingArguments* in the defeat status assignment *UndefeatedArguments* (*DefeatedArguments*) if there is a triggered active defeater *ChallengingArguments* [*ChallengedArguments*], such that

1. *ChallengedArguments* contains *Argument*, or
2. *ChallengedArguments* contains an initial or broadening of *Argument*.

In the first case, the argument *Argument* is *directly defeated* by the arguments *ChallengingArguments*; in the second case, *indirectly*.

We finally have arrived at the formal definition of argumentation stages.

Definition 19.

An *argumentation stage* of an argumentation theory (*Language*, *Rules*, *DefeaterSchemes*) is a defeat status assignment of a range of arguments in the language *Language* with rules in *Rules*,

UndefeatedArguments (*DefeatedArguments*),

such that the following holds:

Argument is an element of *DefeatedArguments* if and only if *Argument* is defeated by arguments that are elements of *UndefeatedArguments*.

The premises and conclusions of the arguments in the range of the argumentation stage are the *premises* and the *conclusions* of the argumentation stage, respectively. The conclusions of arguments in the range of the argumentation stage that are not an initial of another argument in the range, are the *final conclusions* of the argumentation stage. The conclusions of the arguments in *UndefeatedArguments* are the *justified conclusions* of the argumentation stage; the conclusions of arguments in *DefeatedArguments* the *unjustified conclusions*.

The constraint says that the arguments in *DefeatedArguments* are exactly the arguments that are (directly or indirectly) defeated. It turns out that a given range of arguments can correspond to zero, one or several argumentation stages of a theory. Section 6 contains examples.

Our stages are similar to Vreeswijk's (1991, 1993) argument structures. On a formal level, the definitions differ since the approaches to defeat in Vreeswijk's model and in CumulA are different (see chapter 6, section 4). Moreover, the intuitions behind Vreeswijk's arguments structures and CumulA's argumentation are different: Vreeswijk's argumentation structures represent the arguments that are currently undefeated, while CumulA's stages represent both the currently undefeated arguments and the currently defeated arguments. Verheij (1995b, c) argues that the latter is more general and closer to the idea of gradually taking arguments into account.

Verheij (1996a) investigates the relations of CumulA's stages (in a restricted form) and Dung's (1993, 1995) admissible sets of arguments. As Verheij (1996a) shows, there are close relationships on the formal level. However, Dung's admissible sets are seemingly not meant to model stages of the argumentation process. Verheij (1996a) gives examples and formal relations that show that the stages approach generalizes the admissible sets approach, and models the intuition of gradually taking arguments into account.

5 Lines of argumentation and argumentation diagrams

We consider argumentation as a process, in which arguments are taken into account, and are assigned a defeat status. Now that we have described the stages of this process, we will discuss lines of argumentation, that are intuitively series of consecutive argumentation stages.

We treat the construction of arguments in a line of argumentation and the change of status of arguments in a line of argumentation. We finish with the formal definition of lines of argumentation and argumentation diagrams.

5.1 Construction of arguments

In a line of argumentation, arguments are gradually constructed. Since we consider a line of argumentation as a sequence of argumentation stages, the gradual construction of arguments means that the range of the stages in a line of argumentation gradually changes. We distinguish six elementary ways to construct new arguments from the arguments taken into account at some stage, leading to a new stage. We also mention how these constructions affect the premises and conclusions of the stage.

First, at any stage in a line of argumentation a *new statement* can be introduced. Moreover, a line of argumentation can start with a statement. For instance, the initial statement might be:

It is raining.

As mentioned earlier (e.g., in section 2 on arguments), we treat statements as arguments with trivial structure. At this stage of the line of argumentation, where only the statement ‘It is raining’ is taken into account, we have one premise and one conclusion that coincide, namely ‘It is raining’. In general, if at some stage a new statement is introduced, at the new stage a (coinciding) premise and conclusion are added to those of the original stage. [27]

Second, a *forward step* can be added to an argument taken into account. This means that the conclusion of the argument is used to support a new conclusion. For instance, the statement that it is raining can be used to support whether to put on a raincoat or not. We obtain the following single-step argument:

It is raining.
So, it is wise to put on a raincoat.

If a forward step is added to an argument, the premises do not change, but a new conclusion is introduced. In the example, the new conclusion is ‘It is wise to put on a raincoat’.

Third, a *backward step* can be added to an argument. This means that the premise of the argument is supported by a new premise. For instance, if I am in a room that has no windows, I might not take the statement that it is raining for

[27] It can of course be the case that such a premise or conclusion is not new because it was already a premise or conclusion of *another* argument taken into account.

granted, and look for support of the conclusion that it is raining. For instance, the following single-step argument can support that conclusion:

A colleague is completely soaked and tells that it is raining.
So, it is raining.

If a backward step is added to an argument, a premise is replaced by one or more new premises, while the conclusions remain the same. In the example, 'It is raining' is no longer a premise, and is replaced by the premise 'A colleague is completely soaked and tells that it is raining'.

Fourth, a *broadening step* can be added to an argument. This means that the conclusion of a (non-trivial) argument is supported by an additional reason. For instance, it might be the case that the conclusion that it is raining gets additional support by the weather report on the radio. In that case, the previous argument can be broadened to the following argument:

A colleague is completely soaked and tells that it is raining; The weather-report on the radio says that is raining.
So, it is raining.

If a broadening step is added to an argument, the conclusions of the original stage remain the same, while new premises are introduced. In the example, 'The weather-report on the radio says that is raining' is a new premise.

Fifth, two arguments can be combined by *subordination* if one of the arguments taken into account has a premise that is the conclusion of the other. In this way, an argument taken into account can be used to support the premise of another argument. For instance, the argument

A colleague is completely soaked and tells that it is raining.
So, it is raining.

can be subordinated to the argument

It is raining.
So, it is wise to put on a raincoat.

This results in the argument

A colleague is completely soaked and tells that it is raining.
So, it is raining.
So, it is wise to put on a raincoat.

In a case of subordination, a premise and a conclusion of the original stage can be dropped at the new stage.[28] In the example, the premise ‘It is raining’ is dropped.

Sixth, two arguments can be combined by *coordination* if they have the same conclusion. For instance, the arguments

> A colleague is completely soaked and tells that it is raining.
> So, it is raining.

and

> The weather-report on the radio says that is raining.
> So, it is raining.

can be coordinated, resulting in the argument

> A colleague is completely soaked and tells that it is raining; The weather-report on the radio says that is raining.
> So, it is raining.

In a case of coordination, the premises and conclusions of the original stage remain the same at the new stage.

Summarizing, we distinguished six types of argument construction:

1. Introducing a new statement
2. Adding a forward step
3. Adding a backward step
4. Adding a broadening step
5. Subordinating one argument to another
6. Coordinating two arguments

Each of these types has an inverse, that can be considered as a type of argument deconstruction. For instance, the inverse of the introduction of a statement is the withdrawal of a statement. However, we focus on argument construction.[29]

5.2 Change of status

Argumentation stages are characterized by the arguments taken into account and by their status. It is characteristic for argumentation with defeasible arguments that the status of arguments can change in a line of argumentation.

---

[28] It can of course be the case that such a premise or conclusion is not dropped because it is still a premise or conclusion of *another* argument taken into account (cf. note 27).

[29] Technically, as we will see, we will define lines of argumentation in terms of argument construction. Argument deconstruction can be considered as *backtracking* in a line of argumentation.

A basic example of the change of status is reinstatement. In a case of reinstatement, an argument is undefeated at some stage, defeated at a second, later stage, and again undefeated at a third, again later stage. For instance, the argument

The_object_looks_red → The_object_is_red

can first be undefeated, then defeated by the statement

Ralph_says_the_object_is illuminated_by_red_light,

and again undefeated by the statement

Ralph_is_a_liar.

Reinstatement depends on the order in which arguments are taken into account. For instance, if in some line of argumentation the statement that Ralph is a liar was taken into account first, the argument that the object is red would not become defeated.

If we abbreviate the three arguments above as α, β and γ, respectively, all lines of argumentation, corresponding to the six orders in which the three arguments can be taken into account, can be summarized in a so-called argumentation diagram (Figure 1). The nodes in the diagram correspond to argumentation stages. The 0 corresponds to the stage with empty range, at which no arguments have been taken into account. If an argument is defeated in a stage, it is denoted in brackets. The arrows indicate the transition from one stage to the next in a line of argumentation.

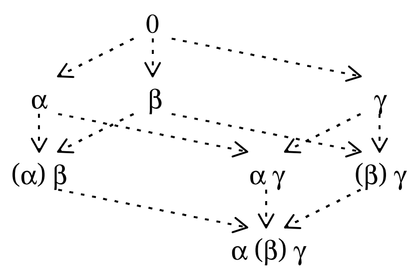

Figure 1: Reinstatement

The diagram shows that in only one of the lines of argumentation the argument α is reinstated, namely in the line of argumentation in which first α, second β, and third γ is taken into account.

In a line of argumentation, the status of an argument can change again and again. This leads to the notion of the status of an argument ‘in the limit’. If in a line of argumentation from some stage onwards the status of an argument remains constant, either undefeated or defeated, the argument is said to be undefeated or defeated in the limit, respectively.[30]

5.3 Definition of lines of argumentation and argumentation diagrams

Shortly we define lines of argumentation and argumentation diagrams. We need some auxiliary definitions.

In order to capture the six ways of argument construction that we discussed, we observe that they have a common characterizing property: the structure of the initial arguments is reflected in the structure of the newly constructed argument. The structural reflection of an argument in another is made precise in the following definition.

Definition 20.

The *maximal argument scheme* of an argument is defined recursively as follows:

1. The maximal argument scheme of an argument of the form *Sentence* is $*$*Sentence*.
2. The maximal argument scheme of the form $\{\{Argument_1, ..., Argument_n\}\} \rightarrow Conclusion$ is $\{\{MaxArgScheme_1, ..., MaxArgScheme_n\}\} \rightarrow Conclusion$, where $MaxArgScheme_i$ is the maximal argument scheme of $Argument_i$, for all i = 1, ..., n.
3. The maximal argument scheme of $\{Arguments_1, ..., Arguments_n\} \rightarrow Conclusion$ is $\{MaxArgSchemes_1, ..., MaxArgSchemes_n\} \rightarrow Conclusion$, where $\{MaxArgSchemes_i\} \rightarrow Conclusion$ is the maximal argument scheme of $\{Arguments_i\} \rightarrow Conclusion$, for all i = 1, ..., n.

An argument *Argument* is *structurally reflected* in an argument *Argument*' if there is an argument in the range of *Argument*' that is an instance of the maximal argument scheme of *Argument*.

The maximal argument scheme is just the argument itself, but with ‘wildcarded premises’. The term ‘maximal argument scheme’ is used because the maximal argument scheme of an argument is the argument scheme that has a (the) maximal set of instances among the argument schemes that have the argument as an instance.

We can now define the successors of a stage, lines of argumentation and argumentation diagrams. The following definition implicitly refers to an argumentation theory (*Language*, *Rules*, *DefeaterSchemes*).

Definition 21.

A stage $Stage_1$ has a stage $Stage_2$ as its *successor* if all arguments in the range of $Stage_1$ are structurally reflected in a stage $Stage_2$. A *line of argumentation* is

[30] In Pollock’s Theory of Defeasible Reasoning (Pollock, 1987-1995) and Vreeswijk’s Abstract Argumentation Systems (Vreeswijk, 1991, 1993) a similar notion is defined.

a sequence of argumentation stages $Stage_1$, $Stage_2$, ..., $Stage_n$, ... (not necessarily finite), such that for all natural numbers i $Stage_{i+1}$ is a successor of $Stage_i$. A *line of forward argumentation* is a line of argumentation that consists of stages with a constant set of premises. A *line of backward argumentation* is a line of argumentation that consists of stages with a constant set of conclusions. An *argumentation diagram* is a set of lines of argumentation.

In a line of argumentation, there is no constraint on the status of the arguments.[31] A stage can have zero, one or several successors. In fact, stages will often have many successors.

Our definitions of successors and lines of argumentation are related to Vreeswijk's definition of successors and argumentation sequences, respectively. However, they differ in three ways. First, the approaches to defeat in Vreeswijk's model and in CumulA are different (see chapter 6, section 4). Second, Vreeswijk's argumentation sequences represent how the set of arguments that are currently undefeated changes in argumentation, while CumulA's lines of argumentation represent how the set of arguments taken into account, whether undefeated or defeated, changes, and how the statuses of the arguments change. Third, CumulA's lines of argumentation are more general than Vreeswijk's argumentation sequences since the latter have fixed premises. Vreeswijk's argumentation sequences are therefore comparable to CumulA's forward lines of argumentation. The relation between successors in Vreeswijk's argumentation sequences is simpler than the relation between successor stages in CumulA's forward lines of argumentation. This is due to the fact that CumulA's stages are representations of all arguments currently taken into account, whether undefeated or defeated, while Vreeswijk's argument structures are only representations of the arguments currently undefeated. The advantages and disadvantages of the two approaches deserve further study.

We stress that the definition of stages above is not a constructive definition of the successors of a stage. It does provide a construction of the *arguments* in the ranges of the successor stages, but not of the *statuses* of these arguments. It is probably not easy to define the relation between the statuses of the arguments in the range of a stage and in the range of a successor, since a change of status of one argument can affect the status of a cascade of other arguments.

Nevertheless, CumulA's lines of argumentation represent how argumentation with defeasible arguments proceeds. More precisely, they represent how argumentation *can* proceed, and not how such argumentation *should* proceed. We give an example of the distinction: both a line of argumentation in which counterarguments are systematically neglected, and one in which at each stage

[31] Henry Prakken has pointed out to me that in cases of multiple stages with equal range, a constraint on the status of arguments seems appropriate. Since each of the multiple stages represents a choice of status, it seems natural that the choice should be kept constant in the successor stages. The problem is that the choice cannot always be kept constant. As a result, it should be made precise how the choice can be kept 'as constant as possible'. We leave this problem for future research.

arguments are challenged by counterarguments that are newly taken into account, fit in the definition above. The second seems closer to how argumentation should proceed. However, a line of argumentation of both types can serve a purpose. A line of argumentation of the first type can help to find all arguments supporting a fixed point of view, while one of the second type can lead more efficiently to justified conclusions.

Which lines of argumentation are preferred with respect to specific purposes and standards, e.g., efficiency, can be regarded as constraints on lines of argumentation. Such constraints define argumentation protocols. Because of the generality of CumulA's lines of argumentation, it seems likely that very different protocols can be defined on them. Research on protocol in the context of argumentation with defeasible arguments has only recently started (see note 6), and is a promising direction of future research.

We finish this section with the definition of forward and backward extensions. Intuitively, a forward extension is the result of collecting as many arguments as possible from a given set of premises. A backward extension is the result of collecting as many arguments as possible, supporting a given set of conclusions.

Definition 22.

A *forward extension* of a set of sentences *Premises* is an argumentation stage *UndefeatedArguments* (*DefeatedArguments*) with premises in *Premises* that has no successor stage with premises in *Premises*. A forward extension *UndefeatedArguments* (*DefeatedArguments*) of a set of sentences *Premises* is *complete* if its range contains all arguments with premises in *Premises*. A *backward extension* of a set of sentences *Conclusions* is an argumentation stage *UndefeatedArguments* (*DefeatedArguments*) with conclusions in *Conclusions* that has no successor stage with conclusions in *Conclusions*. A backward extension *UndefeatedArguments* (*DefeatedArguments*) of a set of sentences *Conclusions* is *complete* if its range contains all arguments with conclusions in *Conclusions*.

A set of sentences can have zero, one, or several forward and backward extensions (possibly with empty range).

The definition of forward extensions has counterparts in many argumentation models, but the distinction between forward and backward extensions is to our knowledge new. Formally our definitions of extensions and complete extensions are close to Dung's (1993, 1995) preferred and stable extensions, respectively.[32] Verheij (1996a) shows the formal relations between Dung's definitions and our definitions (for a version of CumulA, restricted to unstructured arguments and simple defeaters). It turns out that there are subtle distinctions and

[32] Since Dung (1993, 1995) considers unstructured arguments, there is no distinction between forward and backward extensions.

that the definition of extensions above corresponds somewhat better to the intuition that in an extension as many arguments are taken into account as possible.

6 Examples

In this section, we discuss a number of examples of argumentation theories in CumulA. The examples are meant as an illustration of the formal definitions of CumulA.

6.1 Sequential weakening and parallel strengthening

In the sections 3.3 and 3.4, we discussed examples of sequential weakening and parallel strengthening. Here we describe the argumentation theories corresponding to the examples.

First, we treat the sequential weakening example about heaps of sand, based on the sorites paradox. The following argumentation theory represents it, for a fixed natural number n:

Language = {Heap(i) | i = 0, 1, 2, ... }
Rules = {Heap(i) → Heap(i + 1) | i = 0, 1, 2, ... }
DefeaterSchemes = {[Heap(i)] | i > 0 }
∪ {[∗Heap(i) → Heap(i+1) → ... → Heap(i+n)] | | i = 0, 1, 2, ... }[33]

Here Heap(0) abbreviates Body_of_sand_is_heap, Heap(1) abbreviates Body_of_sand_minus_1_grain_is_heap, and for each i = 2, 3, ..., Heap(i) abbreviates Body_of_sand_minus_i_grains_is_heap. The rules say that a body of sand that is one grain fewer than a heap is also a heap. The first set of defeater schemes represents that only the original body of sand is considered a heap without further argumentation. The second set of defeater schemes represents that any argument that contains a sequence of n steps of the rule is defeated.[34] The defeater exactly represents that such an argument becomes defeated because it contains too many steps.

The only statement that can be undefeated is

α_0: Heap(0)

---

[33] For convenience, we have left out the brackets { }.
[34] The choice of n determines the ‘risk’ we accept: for n not too large, say ten, in only a few cases a body of sand is wrongly judged a heap, but at the same time in a few cases reasoning can help us to determine that a body of sand is a heap. For n large, say a billion, we will more often wrongly judge a body of sand a heap, but also reasoning can help us more often. This trade-off between making mistakes and achieving the right results is paramount in reasoning with defeasible arguments.

Therefore the only arguments of this theory that might be undefeated are, for i = 0, 1, 2, ... :

α_i: Heap(0) $\rightarrow$ Heap(1) $\rightarrow$ ... $\rightarrow$ Heap(i)

If the arguments α_0, α_1, ... are consecutively taken into account, the resulting line of argumentation is the following sequence of stages:

α_0
α_0 α_1
α_0 α_1 α_2
...
α_0 α_1 ... α_{n-1} (α_n)
α_0 α_1 ... α_{n-1} (α_n α_{n+1})
...

All arguments that contain a sequence of n steps are defeated. The first of these is the argument α_n. As a result, according to this theory, it is justified that the body of sand is a heap if at most n - 1 grains are taken away from the original heap.

If, for some natural number i_0, the conclusion Heap(i_0) could be justified by some other argument than α_{i_0}, the argument could be extended by n - 1 steps. It would be an undefeated argument different from the defeated α_{i_0+n-1}, and thereby justify that the original body of sand minus i_0 + n - 1 grains is a heap.

Second, we treat the parallel strengthening example about punishing John. The following argumentation theory represents it:

Language = {Robbed, Injured, Minor_first_offender, Punished, Not_punished}
Rules = {{Robbed} $\rightarrow$ Punished, {Injured} $\rightarrow$ Punished,
{Minor_first_offender} $\rightarrow$ Not_punished}
DefeaterSchemes =
{{{$*$Minor_first_offender}} $\rightarrow$ Not_punished [{{$*$Robbed}} $\rightarrow$ Punished],
{{$*$Minor_first_offender}} $\rightarrow$ Not_punished [{{$*$Injured}} $\rightarrow$ Punished],
{{$*$Robbed}, {$*$Injured}} $\rightarrow$ Punished
[{{$*$Minor_first_offender}} $\rightarrow$ Not_punished]}
$\cup$ {[Punished], [Not_punished]}

The three rules say that John is punished if he has robbed, that John is punished if he has injured someone, and that John is not punished if he is a minor first offender. The first two defeaters represent that any argument that ends in {{$*$Minor_first_offender}} $\rightarrow$ Not_punished rebuts any argument that ends in {{$*$Robbed}} $\rightarrow$ Punished or {{$*$Injured}} $\rightarrow$ Punished. The third defeater represents that any coordinated argument that ends with {{$*$Robbed}, {$*$Injured}} $\rightarrow$ Punished rebuts any argument that ends in {{$*$Minor_first_offender}} $\rightarrow$ Not_punished. The last two defeaters represent that the statements that John is punished and that John is not punished are defeated.

The following are the (non-statement) arguments of this theory:

α_1: {{Robbed}} $\rightarrow$ Punished
α_2: {{Injured}} $\rightarrow$ Punished
β: {{Minor_first_offender}} $\rightarrow$ Not_punished
α_{12}: {{Robbed}, {Injured}} $\rightarrow$ Punished

The arguments α_1 and α_2 are the narrowings of the argument α_{12}. In Figure 2, the main lines of argumentation with these arguments are shown.

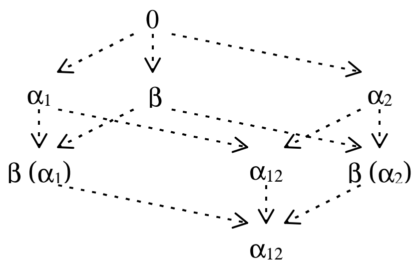

Figure 2: Parallel strengthening

The diagram shows that the arguments α_1 and α_2 only remain undefeated in a line of argumentation if they are both taken into account before β is.

6.2 Conflicting arguments: collective or multiple stages

It is often the case that arguments arise that have incompatible conclusions. Sometimes additional information can be used to resolve the conflict, for instance there can be information about the preference of the arguments.[35] However, it remains possible that there is not sufficient information to resolve the conflict. In that case, the conflict can be resolved by choosing one or more of the arguments involved in the conflict. Two general approaches to dealing with such situations have been proposed in the literature. The first is to discard all arguments in the conflict, as Pollock (1987) does, the second is to discard some of the arguments in such a way that the conflict is resolved, while as few arguments as possible are discarded, as for instance Vreeswijk (1991, 1993) does. Since in the latter case,

[35] In chapter 3, section 6, it is discussed how such conflict-resolving information can be represented in Reason-Based Logic. In chapter 4, section 5, other approaches of dealing with conflicts are treated.

there is normally no unique choice of arguments to discard, multiple solutions can arise.[36]

Both approaches have their merits, and seem reasonable in certain cases.[37] Therefore, in CumulA, both approaches can be dealt with, the first by collective defeat, and the second by multiple stages, i.e., different stages with equal range. As an example, we look at the following arguments:

John has stolen. So, he is punished.
John is a minor first offender. So, he is not punished.
It is nice to have a picnic in the woods. So, we go to the woods.
It is nice to have a picnic at the sea. So, we go to the sea.

The first two of these arguments have incompatible conclusions, the second two also. In the first conflict, it seems best to consider both arguments defeated without further information. In the second conflict, it can be argued that one of the two arguments should be defeated, each choice being equally good. Both are modeled in the following argumentation theory (*Language*, *Rules*, *DefeaterSchemes*):

Language = {Has_stolen, Is_punished, Minor_first_offender, Is_not_punished,
Nice_in_the_woods, Go_to_the_woods, Nice_at_sea, Go_to_the_sea}
Rules = {Has_stolen → Is_punished, Minor_first_offender → Is_not_punished,
Nice_in_the_woods → Go_to_the_woods, Nice_at_sea → Go_to_the_sea}
DefeaterSchemes = {[∗Is_punished, ∗Is_not_punished],
∗Go_to_the_woods [∗Go_to_the_sea], ∗Go_to_the_sea
[∗Go_to_the_woods]}

The main arguments of this theory are:

α: Has_stolen → Is_punished
β: Minor_first_offender → Is_not_punished
γ: Nice_in_the_woods → Go_to_the_woods

36 These solutions correspond to what are often called extensions. In the literature, three perspectives on multiple extensions have been proposed, as Makinson (1994, p. 38) notes: the skeptical perspective, the liberal (or credulous) perspective, and the choice perspective. The skeptical perspective focuses on the intersection of the extensions, the liberal perspective on their union, and the choice perspective on a selected extension. In CumulA, we prefer the latter perspective since the skeptical perspective is closely related to collective defeat, as Pollock (1992, p.7) remarks, which can be dealt with using a compound defeater (cf. section 3.5), while the liberal perspective does not help to resolve conflicts: the union of the multiple extensions that arise to resolve some conflict, again contains the conflict.

37 For instance, Pollock (1994; 1995, pp. 62-64) argues that while in epistemic reasoning unjustified choices are unreasonable, in practical reasoning it is sometimes better to make some choice than none. Since he focuses on epistemic reasoning, he prefers the collective defeat approach.

δ: Nice_at_sea $\rightarrow$ Go_to_the_sea

The two conflicts are handled in different ways: the conflict of the arguments α and β is dealt with by the compound defeater [$*$Is_punished, $*$Is_not_punished], while the conflict of the arguments γ and δ is dealt with by two simple defeaters, $*$Go_to_the_woods [$*$Go_to_the_sea] and $*$Go_to_the_sea [$*$Go_to_the_woods].

Figure 3 shows two argumentation diagrams of this theory. On the left, the arguments α and β are taken into account, and are collectively defeated. On the right, γ and δ are taken into account, resulting in two stages with the same range. (They are separated by a comma.) There are two stages with all four arguments as range, namely (α β) γ (δ) and (α β γ) δ.

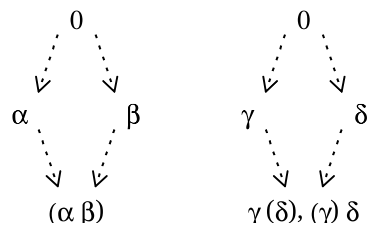

Figure 3: Collective defeat and multiple stages

Although the example argumentation theory is tailor-made for the four mentioned arguments, it shows how general argumentation theories can be defined, in which there is one class of arguments that are collectively defeated in cases of conflict, and another class of arguments that lead to multiple stages in cases of conflict.

To finish the example of collective defeat and multiple stages, we show what happens if there are additional arguments that challenge one of the arguments in the conflict. For instance, there might be two additional arguments

ε: Severe_crime
ζ: Stormy_weather

and two additional defeaters

$*$Severe_crime [$*$Minor_first_offender $\rightarrow$ Is_not_punished]
$*$Stormy_weather [$*$Go_to_the_sea].

In the case of collective defeat and in the case of multiple stages, one of the arguments involved in the conflict is reinstated. Taking into account the argument ε that John's crime was severe, has the effect that α, challenged by ε, is defeated, and that as a result α and β are not collectively defeated. Taking into account the argument ζ that the weather is stormy, has the effect that γ, challenged by ζ, is defeated, and that as a result γ and δ do not give rise to multiple stages. Figure 4 shows the corresponding argumentation diagrams.

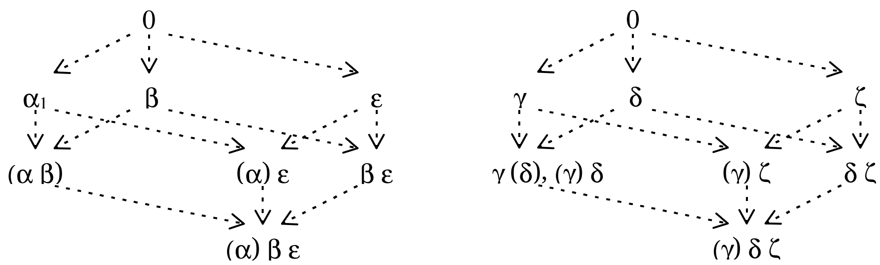

Figure 4: Reinstatement of conflicting arguments

The diagrams shows that collective defeat and multiple stages still occur if ε and ζ are not taken into account.

6.3 Stable marriages

Dung (1995) discusses the so-called *stable marriage problem* in terms of argumentation. In this problem, there is a number of people, some of which love someone else, and some of which are married or, more generally, have a love affair. However love is not always answered, and people do not always have a love affair with the one they love. As a result, love affairs are not necessarily stable. For instance, if John loves Mary, and Mary has a love affair with Alex, the affair of Mary and Alex is in danger, since John will strive for an affair with Mary. However, this threat to Mary and Alex's love affair is overcome if Mary loves Alex: in that case, she will not answer John's attempts. The problem is now to determine which collections of love affairs are stable.[38]

We examine the case that there is a 'love circle': there are n persons person_1, ..., person_n (with n larger than 2), and for i = 1, ..., n, person_i loves person_{i+1}, and person_n loves person_1. In this situation, the fact that person_i loves person_{i+1} is a threat to the affair which person_{i+1} has with person_{i+2}. This case can be translated to an argumentation theory (*Language*, *Rules*, *DefeaterSchemes*), as follows.

Language = {Loves(person_i, person_{i+1}), Affair(person_i, person_{i+1}) | i is an integer modulo n}[39]

Rules = {Loves(person_i, person_{i+1}) → Affair(person_i, person_{i+1}) | i is an integer modulo n}

[38] Dung (1995) discusses the slightly more general problem, in which each person has linearly ordered the other persons according to his or her 'love preference'.

[39] Here 'i modulo n' means 'the remainder of the integer division i/n'.

DefeaterSchemes = {Loves($person_i$, $person_{i+1}$) [$*$Affair($person_{i+1}$, $person_{i+2}$)] | i is an integer modulo n}

We consider the following arguments, for i an integer modulo n:

α_i: Loves($person_i$, $person_{i+1}$) $\rightarrow$ Affair($person_i$, $person_{i+1}$)

These arguments represent that if $person_i$ loves $person_{i+1}$, $person_i$ strives for an affair with $person_{i+1}$.

In the case there are four persons (i.e., n = 4), there are two stable situations, in which all four persons have an affair: either $person_1$ and $person_2$ have an affair, and $person_3$ and $person_4$ have an affair, or $person_2$ and $person_3$ have an affair, and $person_1$ and $person_4$ have an affair. Figure 5 shows the resulting argumentation diagram, for n = 4, that ends in two stages with equal range, that correspond to the two intuitive solutions.

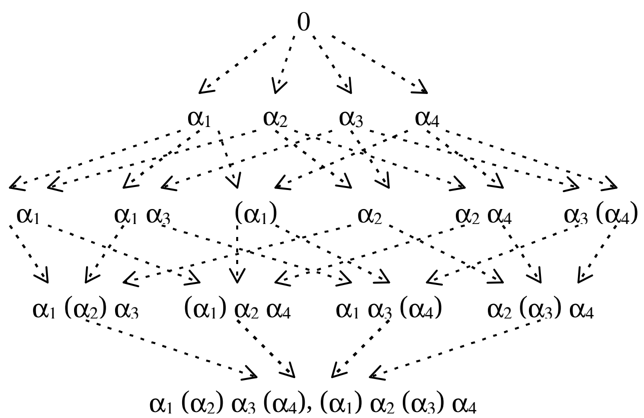

Figure 5: The four-persons case

In the three-persons case (i.e., n = 3), there is no stable solution: any love affair will be threatened.[40] This instability is reflected in the corresponding argumentation diagram (Figure 6). It turns out that there is no stage in which all three arguments are taken into account. Any pair of arguments can be taken into account, but the third argument cannot be. In the figure this is indicated by three question marks ???.[41]

---

40 Note that for n odd at least one of the love affairs involves two persons of the same sex.

41 The fact that there is no stage with all three arguments corresponds in Dung's (1995) approach to the fact that there is no stable extension. The stage approach gives more information about the argumentation theory than Dung's approach since there are stages with less than three arguments. See Verheij's (1996a) comparison of the two approaches for details.

The stages in Figure 6 have a meaning in terms of argumentation. For instance, in the stage α_1 (α_2) the argument α_1 is not challenged, since the argument α_3 is not yet taken into account. The argument α_2 is challenged by Loves(person$_1$, person$_2$). As a result, α_1 justifies Affair(person$_1$, person$_2$), while α_2 cannot justify Affair(person$_2$, person$_3$).

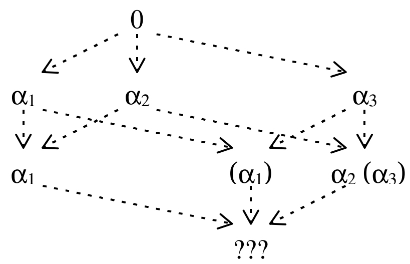

Figure 6: The three-persons case

The three-persons and four-persons cases directly generalize to the cases of any odd and even number of persons, respectively. In the odd case, there is no overall stable solution, in the even case there are two.

6.4 The neurotic fatalist

In the three-persons case above, we saw that not all ranges of argumentation theories correspond to a stage. However, in that example there were maximal subranges that did correspond to a stage, viz. the two-argument subranges. We now show an argumentation theory that has a range, such that there is no maximal subrange that corresponds to a stage.

As an example, we consider the story of the neurotic fatalist. There is one thing our fatalist has been certain of for months: if the world does not end today, it will end tomorrow. Each morning after sunrise he admits that he was wrong the day before, and that the world does not yet end today, but that he nevertheless believes that the world will end the next day.

The arguments of the neurotic fatalist can be formalized in the following argumentation theory:

Language = {World_ends(day$_i$), $\neg$World_ends(day$_i$) | i = 0, 1, 2, ... }
Rules = {$\neg$World_ends(day$_i$) $\rightarrow$ World_ends(day$_{i+1}$) | i = 0, 1, 2, ... }
DefeaterSchemes = {$*\neg$World_ends(day$_i$) [$*$World_ends(day$_j$)] | i > j }

We consider the following arguments, for i any natural number:

α_i: $\neg$World_ends(day$_i$) $\rightarrow$ World_ends(day$_{i+1}$)

At day 0, our fatalist considers the argument α_0 that the world ends at day 1. It is undefeated. The next day he considers the argument α_1: the world did not end at day 1, so it ends at day 2. The argument α_1 defeats the argument α_0. At each new day, he takes a new argument α_i into account, that defeats all previous arguments, since, for each i, the argument α_{i+1} challenges the argument α_i.

We get the following stages if the arguments α_0, α_1, α_2, α_3, ... are consecutively taken into account:

α_0
$(\alpha_0)\ \alpha_1$
$(\alpha_0\ \alpha_1)\ \alpha_2$
$(\alpha_0\ \alpha_1\ \alpha_2)\ \alpha_3$
...

In Figure 7, an overview of these stages is given in an argumentation diagram of the theory.

$$
\begin{array}{c}
0 \\
\downarrow \\
\alpha_0 \\
\downarrow \\
(\alpha_0)\ \alpha_1 \\
\downarrow \\
(\alpha_0\ \alpha_1)\ \alpha_2 \\
\downarrow \\
(\alpha_0\ \alpha_1\ \alpha_2)\ \alpha_3 \\
\downarrow \\
\ldots \\
\downarrow \\
???
\end{array}
$$

Figure 7: The case of the neurotic fatalist

Although the argumentation theory itself may not be considered sensible, the theory is technically interesting since there is no stage with all arguments α_i in its range, nor a maximal subrange that corresponds to a stage. Nevertheless there are several sensible stages. This can be seen as follows.

Assume first that there is a maximal subrange *Subrange*. If *Subrange* is finite, there is a natural number i_0 that is the maximum of the indices i of the arguments α_i in *Subrange*. But then the stage $(\alpha_1\ \alpha_2\ \ldots\ \alpha_{i_0})\ \alpha_{i_0+1}$ has larger range, which contradicts the assumption. Therefore we can assume that *Subrange* is infinite. It is impossible that all arguments α_i in *Subrange* are defeated, since in this argumentation theory an argument can only be defeated by an undefeated argument. Therefore, let i_0 be the smallest natural number i, such that α_i is not defeated. However, if α_{i_0} is not defeated, all arguments that challenge it, i.e., all α_i with $i > i_0$, must be defeated. But that is impossible, since then for each argument α_i there must be an undefeated argument that challenges α_i, and such an argument must have an index larger than i. This contradicts the choice of i_0.

Version June 21, 1999