ARGUE! - AN IMPLEMENTED SYSTEM FOR COMPUTER-MEDIATED DEFEASIBLE ARGUMENTATION

Bart Verheij

Department of Metajuridica
Universiteit Maastricht
P.O. Box 616
6200 MD Maastricht
The Netherlands

+31 43 3883048
bart.verheij@metajur.unimaas.nl
http://www.metajur.unimaas.nl/~bart/

An abbreviated version of this paper has been presented at the NAIC '98 conference:

Verheij, Bart (1998). Argue! - an implemented system for computer-mediated defeasible argumentation. *NAIC '98. Proceedings of the Tenth Netherlands/Belgium Conference on Artificial Intelligence* (eds. Han La Poutré and Jaap van den Herik), pp. 57-66. CWI, Amsterdam.

ARGUE! - AN IMPLEMENTED SYSTEM FOR COMPUTER-MEDIATED DEFEASIBLE ARGUMENTATION

Bart Verheij

Department of Metajuridica
Universiteit Maastricht
P.O. Box 616
6200 MD Maastricht
The Netherlands

+31 43 3883048
bart.verheij@metajur.unimaas.nl
http://www.metajur.unimaas.nl/~bart/

Abstract

This paper introduces the Argue!-system. It is an example of a system for computer-mediated defeasible argumentation, a new trend in the field of defeasible argumentation. In this research, computer systems are developed that can be used to mediate the process of argumentation of one or more users. Argument-mediation systems should be contrasted with systems for automated reasoning: the latter perform reasoning tasks for users, while the former play the more passive role of a mediator. E.g., mediation systems keep track of the arguments raised and of the justification status of statements.

The argumentation theory underlying the system is based on CumulA, a procedural model of argumentation with arguments and counterarguments. The defeasibility of arguments is modeled in terms of argument structure and the attack relation between arguments, and completely independent of the underlying language. The process-model is free, in the sense that it allows not only inference (i.e., 'forward' argumentation, drawing conclusions), but also justification (i.e., 'backward' argumentation, adducing reasons).

In the paper, the Argue!-system is presented from a 'bird's eye' view by focusing on the central ideas. The central user actions of the Argue!-system are inference, justification and attack. Conclusions are inferred from premises, issues are justified by adducing reasons, and arguments are attacked by counterarguments.

Examples are given of the use of the Argue!-system as a philosophical tool. First, the question is raised whether the well-known difference between undercutters and rebutters still stands if defeat is purely interpreted in terms of argument structure (and answered positively). Second, it is argued that the indirect defeat of an argument (e.g., resulting from the defeat of an 'initial part' of the argument) should be terminologically distinguished from defeat by an undercutter. Third, the behavior of attack loops of even and odd length is studied.

The paper ends with a comparison of the Argue!-system with existing systems, and a discussion of the relevance of computer-mediated defeasible argumentation.

1 Computer-mediated defeasible argumentation

Defeasible argumentation can be regarded as a process of *inference, justification* and *attack*. When people argue, conclusions are inferred from premises, premises are justified by adducing reasons, and arguments are attacked by counterarguments.

The understanding of defeasible argumentation has recently been significantly enhanced. Many researchers have devoted attention to the subject. The contributions of Reiter [1980], Pollock [1987, 1994, 1995], Loui [1991, 1992], Vreeswijk [1993, 1997], Dung [1993, 1995], and Hage [1993, 1997] are among the most significant. Reiter's default logic has anticipated several of the ideas now central in defeasible argumentation, and can be considered as a system of defeasible argumentation *avant la lettre*.[1] Pollock has distinguished types of defeat, viz. by undercutters and by rebutters. Loui has emphasized the importance of process and disputation for defeasible argumentation. Vreeswijk has made the study of formalism much more fruitful by using an abstract, unstructured[2] language. Dung has made the important step of studying the formal properties of the notion of attack. Hage has stressed the key role of rules and reasons.

Unfortunately, there is still no uniform terminology in the field, and no 'canonical' formalism modeling defeasible argumentation exists. Nevertheless, even though the present formalisms can have a very different 'feel', there seems to be a convergence of philosophical views. On workshops devoted to defeasible argumentation[3], researchers seem to believe that, despite differences of opinion, a shared body of ideas is arising.

In my dissertation [Verheij, 1996b], I have proposed 'yet another' formalism modeling defeasible argumentation, called CumulA, and attempted to justify it in terms of explicit intuitions. The CumulA-model is subsequently used to give a unifying philosophical perspective on other formalisms modeling defeasible argumentation. Two distinguishing features of the CumulA-model are the following. First, the defeasibility of arguments is fully modeled in terms of argument structure and the attack relation between arguments. It is therefore completely independent of the underlying language, allowing a uniform view on forms of defeat. Second, its process-model is free, in the sense that it allows not only inference (i.e., 'forward' argumentation, drawing conclusions), but also justification (i.e., 'backward' argumentation, adducing reasons). It therefore integrates proof-based systems (that focus on inference) and issue-based systems (that focus on justification). An overview of the CumulA-model is given in section 2.

A relatively recent trend in the field of defeasible argumentation is the development of *computer-mediated defeasible argumentation*. In this research, computer systems are developed that can be used to mediate the process of argumentation of one or more users. The systems can mediate the process in which arguments are drafted and generated by the users, e.g., by

- administering and supervising the argument process,
- keeping track of the issues that are raised and the assumptions that are made,
- keeping track of the reasons adduced and the conclusions drawn,
- keeping track of the counterarguments that have been adduced,
- evaluating the justification status of the statements made, and
- checking whether the users of the system obey the pertaining rules of argument.

Several experimental systems for the mediation of defeasible argumentation have been developed (e.g., IACAS by Vreeswijk [1995], Room 5 by Loui *et al*. [1997], Zeno by Gordon and Karacapilidis [1997], and DiaLaw by Lodder [1998]).[4] The systems differ on their goals, the underlying argumentation theories, and the user interfaces.

[1] As the work of Lin [1993] and Dung [1993] show, Reiter's default logic can to a great extent be rewritten in terms of defeasible argumentation. Moreover, Reiter's semi-normal and normal default rules correspond closely to Pollock's [1987] undercutting and rebutting defeat, respectively (see, e.g., Prakken [1993]).

[2] Well, *almost* unstructured: Vreeswijk's abstract language contains an element denoting contradiction.

[3] For references, see http://www.metajur.unimaas.nl/~bart/argument.htm.

[4] This is only a selection, guided by two criteria: 1. the system must be meant for argument mediation, and 2. argumentation must be defeasible. Not mentioned are, for instance, Nute's [1988] d-Prolog, Pollock's [1995] OSCAR, Tarski's World by Barwise and Etchemendy (see http://csli-www.stanford.edu/hp/), and HUGIN (http://www.hugin.dk/).

In this paper, the Argue!-system is described. It is an argument mediation system, the underlying argumentation theory of which is based on the CumulA-model. After a discussion of the elementary functionality of the Argue!-system (section 3), the Argue!-system is used as a philosophical tool. The abstractness of the system enables experimenting with different forms of defeat and attack-relations. In this way, it can be used to clarify conceptual distinctions and shows typical properties of defeasible argumentation (section 4).

First, the question is raised whether the well-known difference between undercutters and rebutters still stands if defeat is purely interpreted in terms of argument structure (and answered positively) (section 4.1). Second, it is argued that the indirect defeat of an argument (e.g., resulting from the defeat of an 'initial part' of the argument) should be terminologically distinguished from defeat by an undercutter (section 4.2). Third, the behavior of attack loops of even and odd length is studied (section 4.3).

In order to put the Argue!-system in context, it is briefly compared to some other systems for argument mediation (section 5). The paper concludes with a discussion of the relevance of computer-mediated defeasible argumentation (section 6).

2 CumulA: a model of defeasible argumentation in stages

CumulA [Verheij, 1996b] is a procedural model of argumentation with arguments and counterarguments. It is based on two main assumptions. The first assumption is that argumentation is a *process* during which arguments are constructed and counterarguments are adduced. The second assumption is that the arguments used in argumentation are *defeasible*, in the sense that whether they justify their conclusion depends on the counterarguments available at a stage of the argumentation process. If an argument no longer justifies its conclusion it is said to be defeated. The defeat of an argument is caused by a counterargument (that is itself undefeated).

For instance, if a colleague entering the room is completely soaked and tells that it is raining outside, one could conclude that it is necessary to put on a raincoat. The conclusion can be rationally justified, by giving *support* for it. E.g., the following *argument* could be given:

A colleague entering the room is completely soaked and tells that it is raining.
So, it is probably raining.
So, it is necessary to put on a raincoat.

Such an argument is a reconstruction of how a conclusion can be supported.

An argument that supports its conclusion does not always justify it. For instance, if in our example it turns out that the streets are wet, but the sky is blue, the conclusion that it is necessary to put on a raincoat would no longer be justified. The argument has become *defeated*. For instance, the following argument could be given:

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

In this case the argument that it is probably raining is defeated by the *counterargument* that the shower is over. Since the conclusion that it is probably raining is no longer justified, it can no longer support the conclusion that is is necessary to put on a raincoat.

CumulA is a procedural model of argumentation with arguments and counterarguments, in which the defeat status of an argument, either undefeated or defeated, depends on:

(1)the structure of the argument;
(2)the attacks by counterarguments;
(3)the argumentation stage.

We briefly discuss each below. The model especially builds on the work of Pollock [1987, 1995], Vreeswijk [1993, 1997] and Dung [1995] in philosophy and artificial intelligence, and was developed to complement the work on the model of rules and reasons Reason-Based Logic (see, e.g., Hage [1996, 1997] and Verheij [1996b]).

In the model, the structure of an argument is represented as in the argumentation theory of Van Eemeren and Grootendorst [1981, 1987]. Both the subordination and the coordination of arguments are possible. It is explored how the structure of arguments can lead to their defeat. For instance, the intuitions that it is easier to defeat an argument if it contains a longer chain of defeasible steps ('sequential weakening'), and that it is harder to defeat an argument if it contains more reasons to support its conclusion ('parallel strengthening'), are investigated.

In the CumulA-model, which arguments are counterarguments for other arguments, i.e., which arguments can *attack* other arguments, is taken as the primitive notion [cf. Dung, 1995; see also Verheij, 1996a]. This approach to argument defeat can be called *counterargument-triggered defeat*. Basically, an argument is defeated if it is attacked by an undefeated counterargument. This approach to argument defeat must be contrasted with *inconsistency-triggered defeat*: the primitive notion is which arguments have conflicting conclusions (as, e.g., in Vreeswijk's [1993, 1997] abstract argumentation systems). In this approach to argument defeat, an argument is defeated if there is an undefeated argument with conflicting conclusion. Often the defeating argument has higher priority than the defeated argument, with respect to some priority relation on arguments.[5]

In CumulA, so-called *defeaters* indicate which arguments are counterarguments to other arguments, i.e., which arguments can defeat other arguments. In this way, CumulA shows that the defeasibility of arguments can be fully modeled in terms of argument structure and the attack relation between arguments, independent of the underlying language. Moreover, it turns out that defeaters can be used to represent a wide range of types of defeat, as proposed in the literature, e.g., Pollock's [1987] undercutting and rebutting defeat. Also some new types of defeat can be distinguished, namely defeat by sequential weakening (related to the well-known sorites paradox) and defeat by parallel strengthening (related to the accrual of reasons).

In the CumulA-model, argumentation stages represent the arguments and the counterarguments currently taken into account, and the status of these arguments, either defeated or undefeated. The model's lines of argumentation, i.e., sequences of stages, give insight in the influence that the process of taking arguments into account has on the status of arguments. For instance, by means of argumentation diagrams, which give an overview of possible lines of argumentation, phenomena that are characteristic for argumentation with defeasible arguments, such as the reinstatement of arguments, are explicitly depicted. In contrast with Vreeswijk's [1993, 1997] model, we show how in a line of argumentation not only new conclusions are inferred ('forward argumentation', or inference), but also new reasons are adduced ('backward argumentation', or justification). In other words, CumulA's

[5] The distinction of counterargument-triggered and inconsistency-triggered defeat is made in my dissertation [Verheij, 1996b]. I think that (Dung-style) counterargument-triggered defeat is philosophically the most attractive and innovative of the two approaches to argument defeat. One reason is that counterargument-triggered defeat can be modeled completely independent of the language, as in the CumulA-model (see note 2). Another is that one of the central types of defeat, viz. defeat by an undercutter, is not inconsistency-triggered (see section 4.1).

process-model is free, as opposed to proof-based systems (that focus on inference) and issue-based systems (that focus on justification).

To summarize, CumulA shows

(1) how the subordination and coordination of arguments is related to argument defeat;
(2) how the defeat of arguments can be described in terms of their structure, counterarguments, and the stage of the argumentation process, and independent of the logical language;
(3) how both inference and justification can be formalized in one model.

CumulA has obvious limitations. We mention two. First, its underlying language is completely unstructured. It contains for instance no logical connectives, no quantifiers, and no modal operators. This is certainly a limitation, but one of the research objectives was to show that defeat can be fruitfully studied independent of the language. Second, the role of rules in argumentation in CumulA is not clarified. This is in part due to the first limitation: the language of CumulA does not contain a conditional, which can express rules.[6]

Verheij [1996b] discusses the CumulA-model extensively, both informally and formally.

3 The Argue!-system

The Argue!-system is a system for computer-mediated defeasible argumentation with a graphical user interface. The user 'draws' the argumentation data, by clicking and dragging a pointing device, such as a mouse. Its underlying argumentation theory is based on CumulA [Verheij, 1996b].[7]

The following description of the Argue!-system takes a 'bird's eye' view by focusing on central ideas, and does not give a detailed description of the data structures, algorithms and interface of the Argue!-system.

3.1 Inference, justification and attack

Central actions of defeasible argumentation are inference, justification and attack. We show how these can be performed in the Argue!-system.

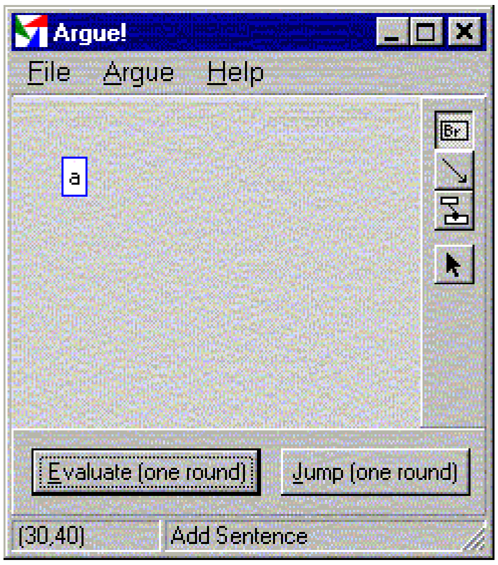
Argumentation starts with making a *statement*. Statements can be of two types: *assumptions* and *issues*. An assumption is taken as justified, regardless of reasons supporting it, or counterarguments attacking it. Issues initially have no justification status; their status depends on the supporting reasons and the attacking counterarguments. In Figure 1 (left), the statement 'a' is made, as an assumption. To indicate that it is an assumption, it is shown in a blue-bordered box.

The first central action of defeasible argumentation is *inference*. Once a statement is made it can be used for inferring a conclusion, say 'b', as in Figure 1 (right). It should be noted that

[6] Verheij [1996b] does contain a formal model in which rules play a central role, viz. Reason-Based Logic. However, the formal connection with the CumulA-model is not made. The cause of this is amongst others the very different 'flavours' of the two formalisms.

[7] The argumentation theory underlying the Argue!-system is however not *equal* to the CumulA-model. I mention five differences. First, the distinction of assumptions and issues does not occur in the CumulA-model. Second, in CumulA, reasons can have subreasons, whereas in the Argue!-system this is not possible. Third, the defeaters of the Argue!-system can (due to their graphical lay-out) not represent CumulA-defeaters in their full generality. Fourth, in the Argue!-system, all argumentation data (sentences, arrows and defeaters) can be added dynamically during a session, whereas lines of argumentation in CumulA use a fixed background theory of arrows (called rules in CumulA) and defeaters. Fifth, whereas CumulA merely gives constraints for sensible evaluations of arguments and statements, the Argue!-system provides two algorithms computing such evaluations in rounds (adjustable at run-time, for research purposes).

the user determines which statements are reasons for other statements, by adding appropriate input. It is for instance not the case that the user only can draw inferences according to the rules of inference as allowed by, let's say, first-order predicate logic.[8] The statement 'b' is added as an issue, which is visualized by the different border color of its box.



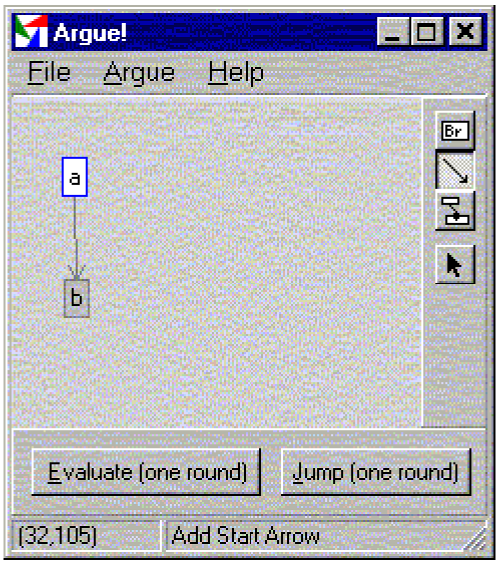


Figure 1

The Argue!-system is an evaluative system for argument mediation: the user provides the argumentation data, such as assumptions, issues, reasons, and (as we will see later) attacks. The system determines the justification status of statements, i.e., whether they are *justified*, *unjustified*, or *neither*.

When the user enters new argumentation data, statements obtain their initial value: assumptions are initially justified, issues are initially neither justified nor unjustified. In Figure 1 (right), 'a' is justified, since it is an assumption, and 'b' is neither justified nor unjustified, since it is an issue. Justified statements are shown in white boxes, unjustified statements (as we will see later) in crossed white boxes, statements that are neither justified nor unjustified in gray boxes.

The Argue!-system has two built-in algorithms that help determining the justification status of arguments: 'evaluate' and 'jump'.[9] We will only discuss the 'evaluate' algorithm. The system evaluates the statements when the user clicks the 'Evaluate'-button. The system evaluates statements in rounds: the justification statuses of statements are used as input for computing their status in the next. The Argue!-system has (among others) the following two evaluation rules:

- If a statement is an assumption, it is justified.
- If a statement is an issue, and has justified support, it is justified.

The result is shown in Figure 2 (left): both statements are justified.

---

[8] Recall the limitations of the CumulA-model, discussed in section 2: argumentation in CumulA is not rule-based.

[9] In order to make experimentation with the Argue!-system as flexible as possible, both algorithms can be adjusted by the user at run-time.

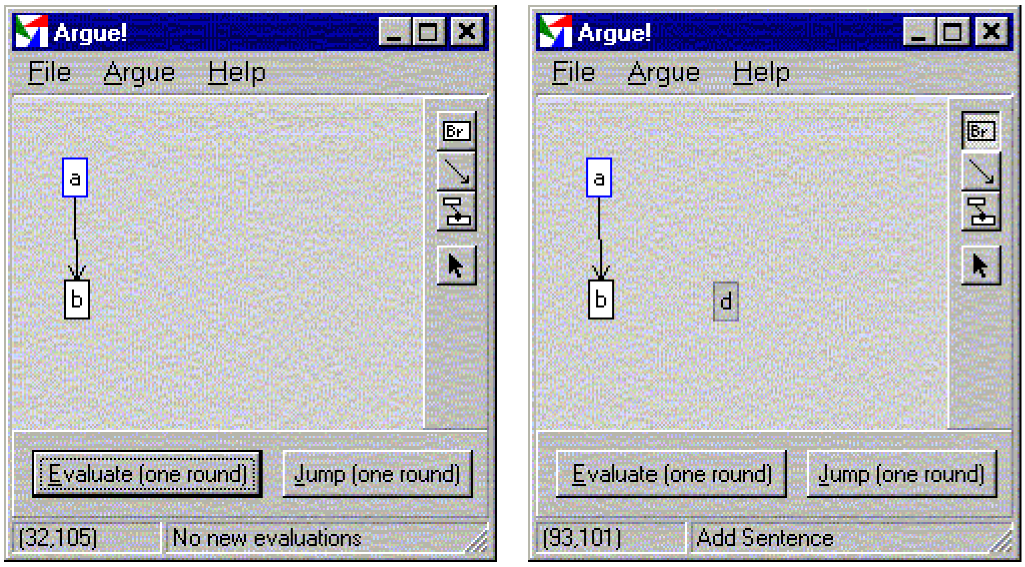


Figure 2

The second central action of defeasible argumentation is *justification*. In Figure 2 (right), a new issue is raised: 'd'. As yet, 'd' is neither justified, nor unjustified. It can be justified by giving justified support for it, in the form of an assumption 'c' (Figure 3, left and right, before and after evaluation, respectively).

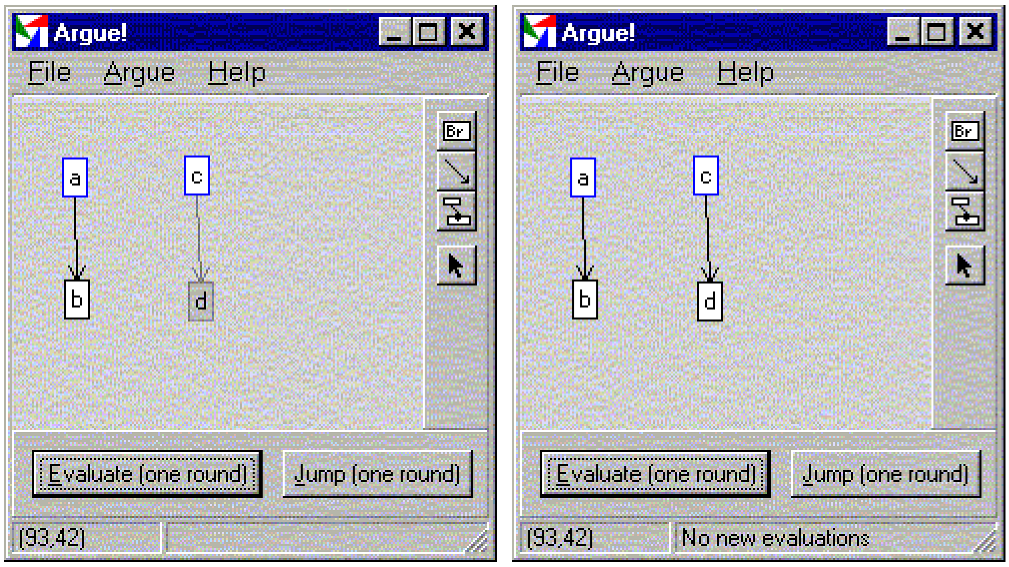


Figure 3

The third central action of defeasible argumentation is *attack*. In Figure 4 (left, before evaluation), the user has added a defeater, visualized by a special visual shape, that consists of two connected rectangles. The argument configuration (i.e., sentences-in-boxes and arrows) contained in the first rectangle is *challenging*, the argument configuration in the second (indicated by the diamond) is *challenged*. In the defeater in Figure 4 (left), 'd' is challenging, and 'a' supporting 'b' is challenged. The defeater represents that 'd' is a counterargument to 'a' supporting 'b': if 'd' is justified, 'a' does not support 'b' and 'b' is unjustified. It should be noted that 'a' is not challenged by 'd'.

In Figure 4 (right), the result of evaluation is shown: 'b' has become unjustified (indicated by the cross), and the arrow between 'a' and 'b' has become dotted, since it is no longer supporting. Since 'd' is justified, we say that 'd' *attacks* 'b' and 'a' supporting 'b'.

The evaluation rule of the Argue!-system corresponding to attack is the following:

- If a statement has no justified support and is attacked, it is unjustified.

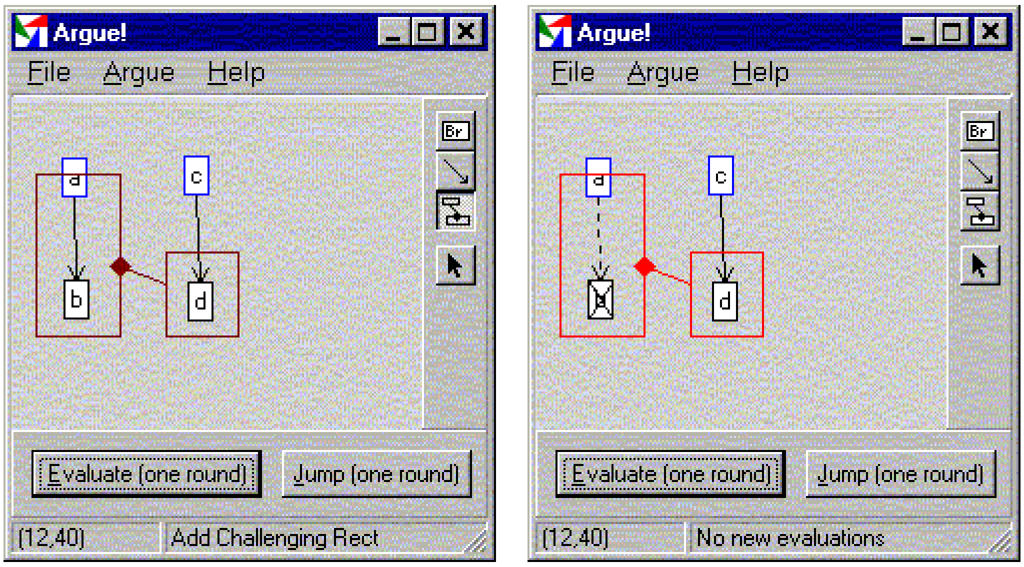
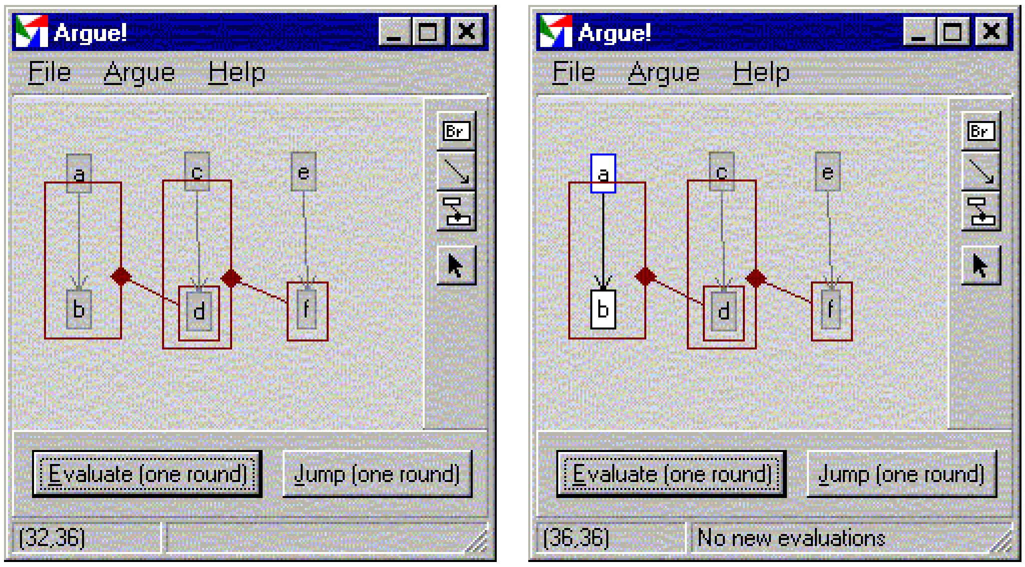


Figure 4

3.2 *Reinstatement*

Reinstatement is typical for defeasible argumentation. An argument is said to be *reinstated* if it becomes undefeated after being defeated after being undefeated. In Figure 5 (left), a configuration is shown that allows the reinstatement of the argument 'a $\rightarrow$ b'. It contains two defeaters, the first representing that 'd' challenges 'a' supporting 'b', and the second that 'f' challenges 'c' supporting 'd'. Initially, all statements 'a' to 'f' are issues, so all arguments are neither undefeated nor defeated, and all statements are neither justified nor unjustified.

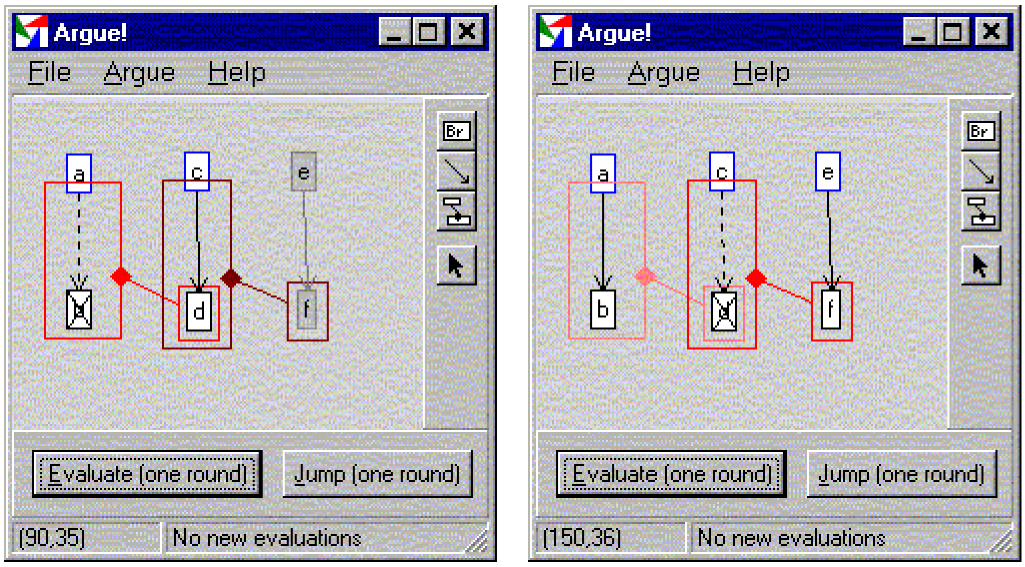
In Figure 5 (right), 'a' is turned into an assumption. The argument 'a $\rightarrow$ b' is undefeated, and 'b' is justified.



(32,36)
(36,36) No new evaluations

Figure 5

Next, 'c' is turned into an assumption (Figure 6, left). As a result, 'd' becomes justified, and attacks 'a' supporting 'b'. The argument 'a → b' has become defeated, and 'b' is unjustified. Finally, 'e' is turned into an assumption (Figure 6, right). The statement 'f' is justified, and makes the argument 'c → d' defeated. As a result, 'd' is unjustified. The argument 'a → b' is *reinstated*: it becomes undefeated again.



(90,35) No new evaluations
(150,36) No new evaluations

Figure 6

4 The Argue!-system as a philosophical tool

In the following, example sessions with the Argue!-system are discussed. They are chosen as illustrations of the use of the Argue!-system as a philosophical tool, that can help to clarify conceptual distinctions and show typical properties of defeasible argumentation.

4.1 Undercutters and rebutters

Pollock [1987] made the distinction between *undercutters* and *rebutters*.[10] Both undercutters and rebutters can defeat *prima facie* reasons. Rebutters are reasons denying the conclusion of another reason, while undercutters attack the connection between the reason and the conclusion rather than attacking the conclusion itself [Pollock, 1987, p. 484/5].[11]

What are the counterparts of undercutters and rebutters in the Argue!-system? The basic example of an attack (Figure 4, right) can be taken as an example of an undercutter: 'd' attacks the connection between the reason 'a' and the conclusion 'b'. In Figure 7 (left), an example of a rebutter is shown: 'c' is a reason denying the conclusion 'b' of 'a', for it is a reason for 'not-b'.[12]

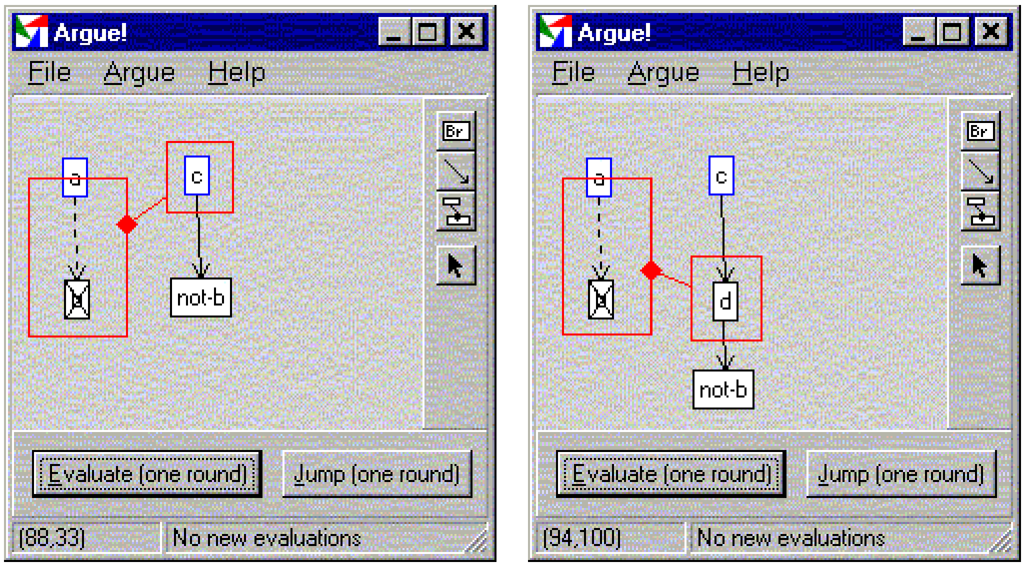


Figure 7

Let's look closer at the two defeaters corresponding to an undercutter (Figure 4, right) and a rebutter (Figure 7, left). Both have the same formal structure! There is one challenging sentence and a conclusion and its connection with a reason is challenged. The only difference is that the rebutter itself supports the conclusion 'not-b'. The point becomes even more clear if the example of an undercutter (Figure 4, right) is turned in an example of a rebutter, simply by making 'd' into a reason for 'not-b' (Figure 7, left). In other words, if defeat is interpreted in terms of argument structure and the attack relation between arguments (as in CumulA), the forms of defeat corresponding to undercutters and rebutters seem to coincide at first sight.

Let's us take a closer look at undercutters and rebutters, and try whether their 'structural conincidence' can withstand a more fine-grained analysis. After all, the nature of undercutters and rebutters might not have been properly represented.

Two points can be made. The first is that an undercutter *only* needs to attacks the connection between a reason and a conclusion (as in Figure 8, left), and not *also* the

[10] Actually he speaks of undercutting and rebutting *defeaters*. Since in this paper the notion 'defeater' has a somewhat different, technical meaning, we speak of undercutters and rebutters.

[11] Recall Verheij's [1996b] distinction between counterargument-triggered and inconsistency-triggered defeat (see note 5). Defeat by an undercutter is a typical example of the first, and by a rebutter of the second.

[12] It should be noted that the Argue!-system does *nothing* with the structure of the sentences adduced as statements. To the Argue!-system, the sentence 'not-b' has no relation to 'b'. In CumulA and the Argue!-system, defeat is modeled in terms of the structure of arguments and counterarguments, and completely independent of the underlying language. Modeling defeat *independent* of the language was one of the design goals of CumulA. See notes 2 and 5.

conclusion (as in Figure 4, right). An undercutter does not necessarily make a conclusion unjustified, but only makes the reason for it not supporting. The second is that a rebutter can *only* attack provided that it supports and justifies[13] its conclusion (as in Figure 8, right). A rebutter is normally thought of as a kind of preference of reasons. The preference criterion might fail to hold if the preferred reason is not justifying. In the case of a rebutter, the conclusion that is *prima facie* supported by the rebutted reason, becomes unjustified, since its opposite is justified.

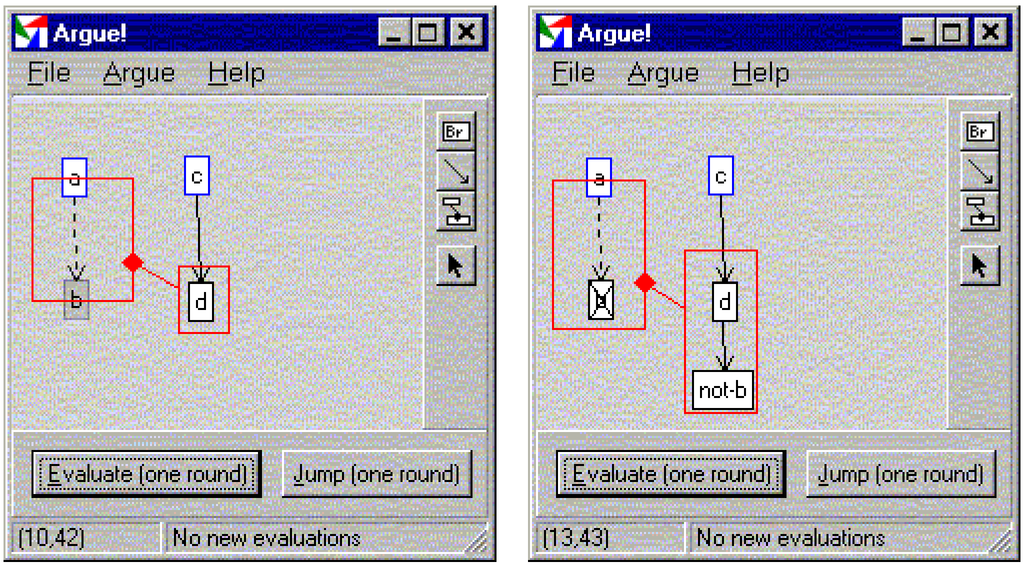


Figure 8

The two defeaters in Figure 8 clearly represent different forms of defeat. The defeater on the left represents that 'a' does not support 'b' if 'd' is justified. The result of the attack by 'd' is no longer that 'b' is unjustified, but only that 'b' is neither justified nor unjustified. The defeater on the right represents that 'a' does not support 'b' and 'b' is unjustified if 'd' supports 'not-b' and 'd' and 'not-b' are both justified.[14] So, on second thoughts, a sensible difference between defeat by an undercutter and by a rebutter can still be made, even if defeat is interpreted in terms of argument structure and the attack relation between arguments.

4.2 Direct and indirect defeat

In defeasible argumentation, arguments can not only be directly defeated, but also indirectly [Verheij, 1996b]. Figure 9 gives an example. On the left, 'c' is justified since it is the conclusion of an undefeated argument, namely the argument 'a → b → c'. The statement 'd' is challenging, but not attacking, since it is adduced as an issue. On the right, 'd' is adduced as an assumption. As a result, 'd' attacks 'a' supporting 'b'. The argument 'a → b' is defeated. Because its defeat is the immediate result of the attack by 'd', we say that 'a → b' is *directly defeated*. There is another argument the conclusion of which is no longer justified, namely the argument 'a → b → c'. Also that argument is defeated. Since its defeat is the result of the defeat of its initial part 'a → b', we say that 'a → b → c' is *indirectly defeated*.

[13] Note that a supported conclusion is not necessarily justified.
[14] In terms of Reason-Based Logic [Hage, 1996, 1997; Verheij, 1996b]: an undercutter corresponds to a reason that makes a rule not apply and not give rise to a reason, while a rebutter corresponds to a reason for a conclusion that outweighs a reason against the conclusion.

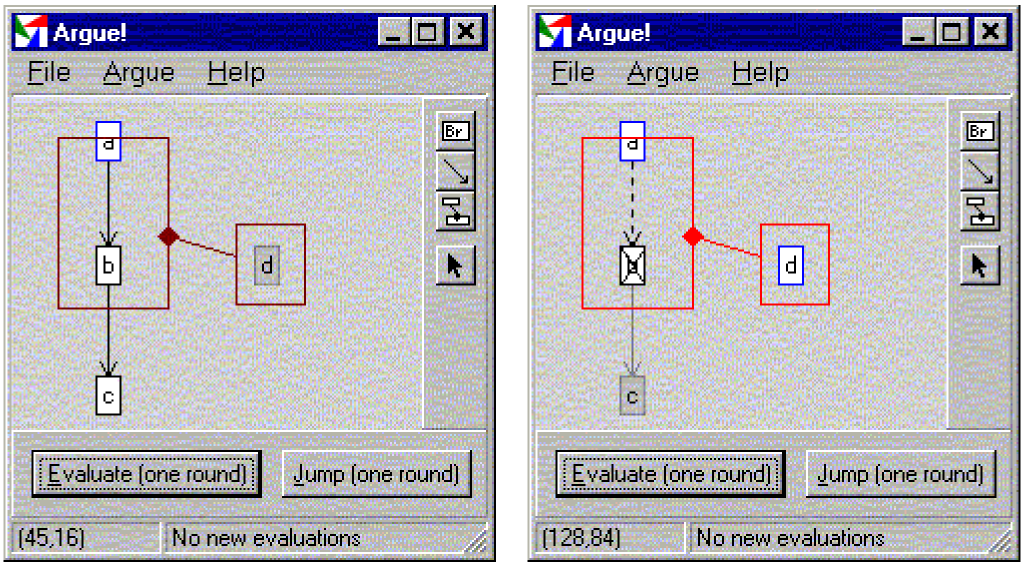


Figure 9

Unfortunately, in the literature a special case of indirect defeat of arguments (viz. indirect defeat by the defeat of the premises of arguments) has also been called undercutting defeat, which suggests a relation to Pollock's [1987] undercutters. The reason for this confusing terminology is maybe that, just as in a case of defeat by an undercutter, in indirect defeat the justifying connection between a reason and conclusion is lost. However, in indirect defeat, the connection between a reason and a conclusion *is not attacked* as in the case of Pollock's [1987] undercutters. There is no direct relation between an attack and the defeat. Instead the justifying connection between a reason and a conclusion is just not 'activated', because the reason is not justified. This is not necessarily caused by an (indirect) attack, but can simply be the result of the fact that the reason is an issue, and not an assumption. See, e.g., Figure 5 (right) where all three connections between a reason and a conclusion are not activated. However, this is not the result of an attack (since there is no attack), but of the fact that all statements are issues.

Clearly, indirect defeat on the one hand and direct defeat by an undercutter on the other are of a very different nature. It is therefore better to use distinguishing terminology.

4.3 Attack loops

In defeasible argumentation, attack loops have interesting properties, that can be studied in the Argue!-system. An *attack loop* is a sequence of arguments A_1, A_2, ..., A_n (for some natural number n, called the *length* of the attack loop), such that A_i attacks A_{i+1} (for i a natural number, such that $1 \leq i < n$) and A_n attacks A_1. The behavior is different for attack loops of odd and of even length (see also Verheij [1996a] and [1996b, p. 151]). We illustrate the difference by discussing an attack loop of length two and three.[15]

An attack loop of even length

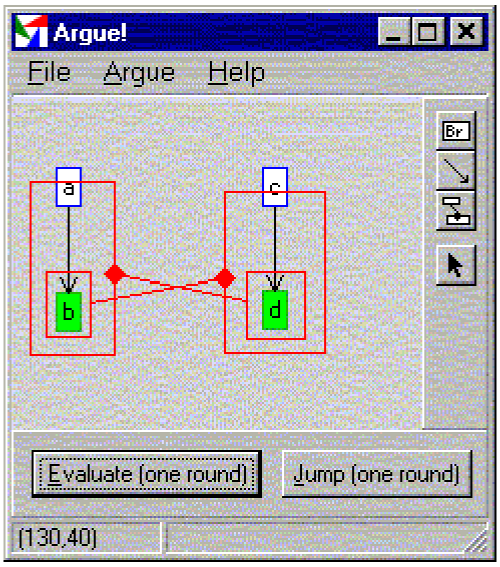
An attack loop of length two is shown in Figure 10. The statement 'b' attacks 'c' supporting 'd', and the statement 'd' attacks 'a' supporting 'b'.

Recall that in the Argue!-system evaluation occurs in rounds: the justification statuses of statements are used as input for computing their status in the next. Until now, this has not had

[15] Similar examples have been studied in the field of nonmonotonic logic.

effects on the system's output. In the case of attack loops, evaluation in rounds makes a difference: evaluation can loop. We explain why.

If in the configuration of Figure 10, both 'a' and 'c' are assumptions, the first round of evaluation will make both 'b' and 'd' justified, since they both have justified support (Figure 10, left). (The statements 'b' and 'd' are in green boxes, since their evaluation status has changed with respect to the previous round.) Now 'b' and 'd' attack each other, so in the second round, both 'b' and 'd' become unjustified (Figure 10, right). Since 'b' and 'd' no longer are attacks, in the next round, 'a' and 'c' are again supporting, resulting in 'b' and 'd' being justified. In short, evaluation flips between the two states on the left and the right of Figure 10.



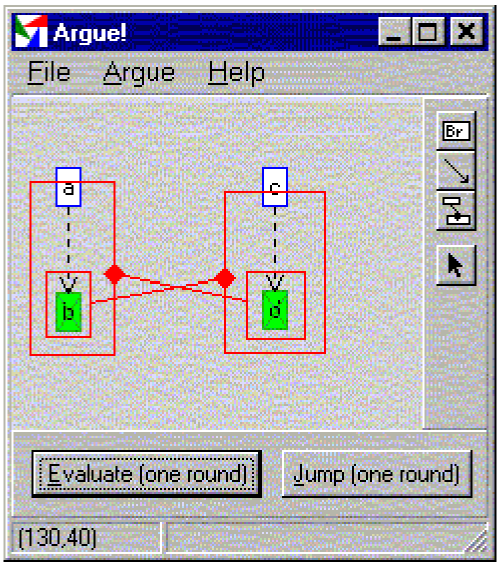


Figure 10

Is there no 'stable' evaluation state? There is: Figure 11 (left) gives one. The statement 'b' is justified since it has justified support, and 'd' is unjustified since it has no justified support and is attacked by 'b'.

There are two stable states. The second is the 'mirror image' of the first.[16] The two stable states can be reached by *manipulating the order* in which the attacks are active. If 'b' attacks 'c' supporting 'b' *before* 'd' can attack 'a' supporting 'b', a stable evaluation is reached. Figure 11 (right) shows how: if at first 'a' is an assumption and 'c' an issue, the 'b'-attack comes first. Then changing 'c' to an assumption, the stable evaluation of Figure 11 (left) is reached.

[16] Attack loops of length two correspond to what is sometimes called the Nixon diamond in nonmonotonic logic. For instance, in Reiter's [1980] default logic, the Nixon diamond leads to two extensions, corresponding to the two stable evaluations here.

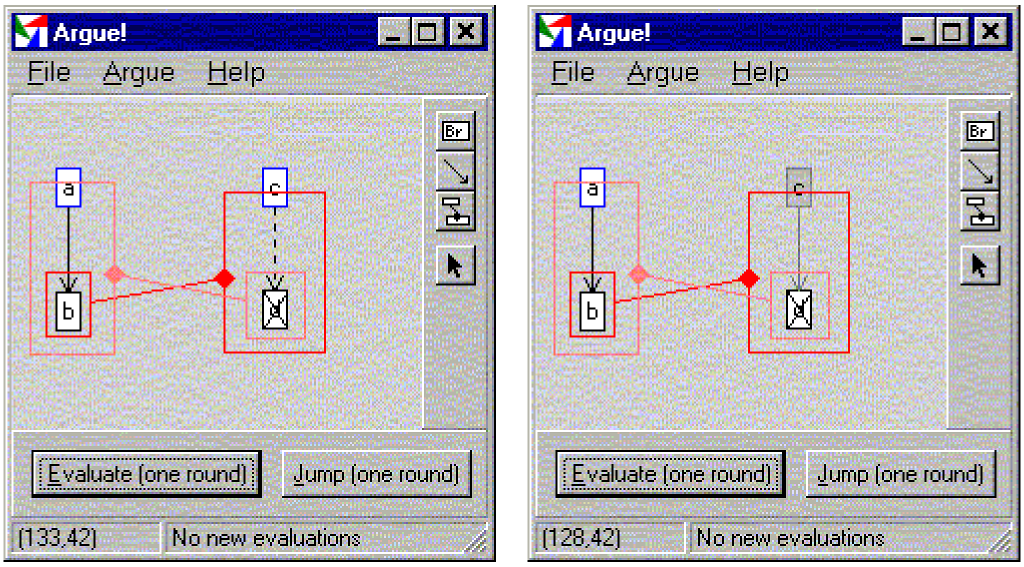


Figure 11

Summarizing, in Figure 10, evaluation looped since both attacks were active or inactive at the same time, while, in Figure 11, a stable evaluation could be reached, by activating one attack before the other.

An attack loop of odd length

An attack loop of length three is shown in Figure 12. The statement 'b' attacks 'c' supporting 'd', the statement 'd' attacks 'e' supporting 'f', and 'f' attacks 'a' supporting 'b'.

If 'a', 'c' and 'e' are adduced as assumptions, evaluation loops analogous to what we saw in the case of a length-two loop, flipping between the two states of Figure 12. Again, the loop occurs because the three attacks are concurrently activated and deactivated.

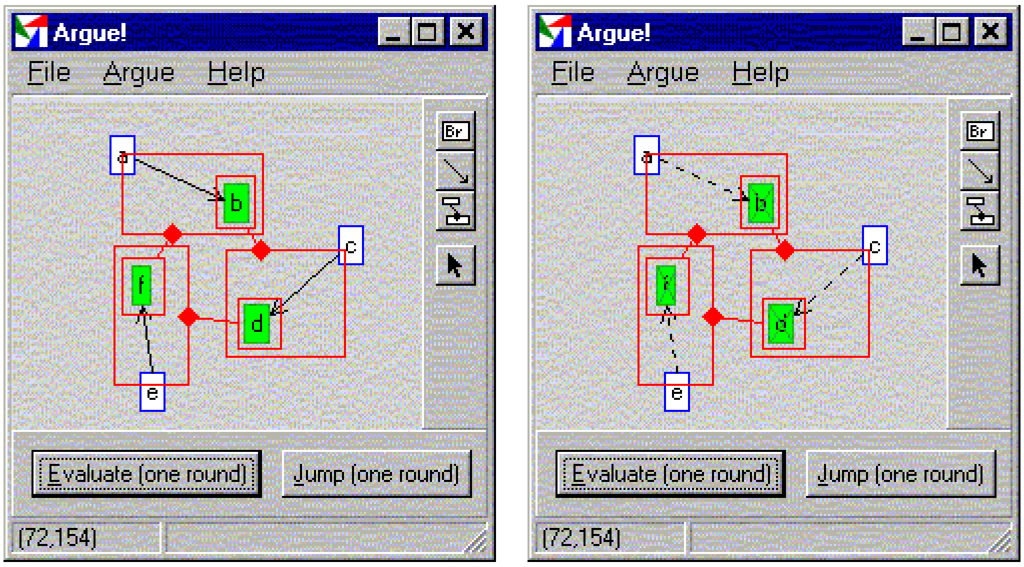


Figure 12

This time, there is no stable evaluation, as can be easily seen.[17]

[17] Assume that 'b' is attacking. Then 'd' cannot be since it is unjustified. But then 'f' is justified and attacking 'b', which implies that 'b' is unjustified. This is impossible since 'b' is attacking. The assumption that 'b' is attacking

Nevertheless, manipulating the order of attacks leads to interesting behavior. If 'b' is made to attack first (as in Figure 13, left), a *different* evaluation loop occurs, of length six. One of its states is shown in Figure 13 (right). The other states are obtained by making the obvious permutations.

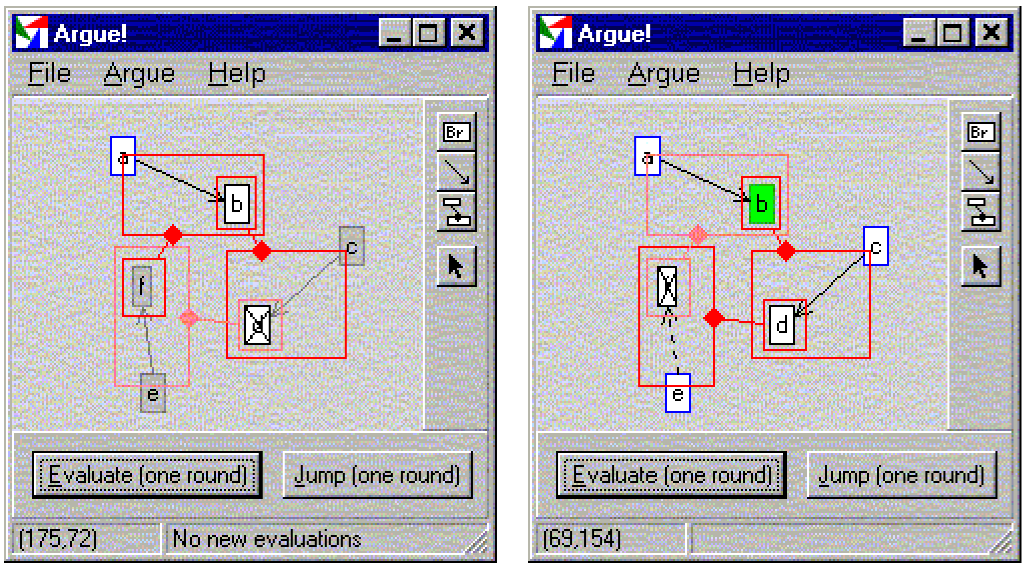


Figure 13

It is not hard to see (using reasoning similar to that of note 17) that the two evaluation loops, one of length 2 (Figure 12), the other of length 6 (Figure 13, right), are essentially the only two.

5 A comparison of argument-mediation systems

In order to put the Argue!-system in context, it is briefly compared to four other systems, namely IACAS by Vreeswijk [1995], Room 5 by Loui *et al.* [1997], Zeno by Gordon and Karacapilidis [1997], and DiaLaw by Lodder [1998].[18]

The goals for building the systems differ. IACAS (which stands for InterACtive Argumentation System) is presented as an implementation of Chisholm's principles of knowledge. Room 5 is called a testbed for public interactive semi-formal legal argumentation. Zeno is meant to create advanced support for complex multi-party/multi-goal decision-making. DiaLaw is a dialog game modeling legal justification.

First, the underlying argumentation theories are discussed; second, the user interfaces.

5.1 The underlying argumentation theories

In the underlying argumentation theories of all four systems argumentation is *dynamic*. Statements can be made, and reasons can be adduced. In IACAS, Room 5, Zeno and DiaLaw, argumentation is *issue-based* (as in Rittel's well-known Issue-Based Information System (IBIS) [Rittel and Webber's, 1973]). No new conclusions can be drawn, since these systems focus only on justification of an initial central issue. In the Argue!-system, argumentation is *free*, in the sense that there is no central issue, and both inference (i.e., 'forward'

must be dropped. Similarly, 'd' and 'f' cannot be attacking. If all three are not attacking, the evaluation loop of Figure 12 arises.

[18] See note 4.

argumentation, drawing conclusions) and justification (i.e., 'backward' argumentation, adducing reasons) are allowed.

All systems model a notion of *defeasibility* of argumentation. IACAS, Room 5, Zeno and DiaLaw have a notion of *reasons for and against conclusions*. In Zeno and DiaLaw, weighing the conflicting reasons determines which conclusions are justified. DiaLaw and the Argue!-system have an *undercutter-type exception*.

IACAS and DiaLaw have a notion of the *rules* underlying argument steps. DiaLaw is based on the theory of rules and reasons Reason-Based Logic (see, e.g., Hage [1996, 1997] and Verheij [1996b]).

In Room 5, Zeno and DiaLaw, argumentation is considered as a *game with participants*. In Room 5 and Zeno, the game character is left *implicit*, but obtained by the distributed access to the systems, on the World-Wide Web. In DiaLaw, the game character is made *explicit* in the form of a dialogue game with two parties. IACAS and the Argue!-system have no explicit notion of participants.

IACAS, Zeno and the Argue!-system are *evaluative*: the status of statements and arguments can be determined by the system.

5.2 The user interfaces

IACAS and DiaLaw have a *text-based interface*; moves are typed at a command-prompt. The Argue!-system has a *graphical interface*: argumentation data are 'drawn' using a pointing device. Room 5 and Zeno have a *template-based interface*: users fill in forms to perform an argument move.

Room 5, Zeno and the Argue!-system present arguments in a *visual* manner. Zeno and the Argue!-system use a *tree*-like presentation. Room 5 uses a clever system of *boxes-in-boxes* in an attempt to avoid 'pointer-spaghetti'. In IACAS and DiaLaw, argumentation is presented in a *verbal* manner, since they have a command-line interface.

In Room 5 and Zeno, counterarguments (formed by reasons against conclusions) are *grouped together* in the visual argument structure. In the Argue!-system, counterarguments are shown by a *special visual structure* (viz. defeaters). In IACAS, the system can generate a list of arguments and counterarguments. In DiaLaw, counterarguments are not directly accessible.

In DiaLaw, the dynamic aspect of argumentation is shown by a *view on the sequence of moves*. In IACAS, Room 5, Zeno and the Argue!-system, only a *view on the current stage* of the argumentation process is visible. In Room 5, it is possible to switch between different views showing different types of information.

5.3 Conclusions

If we look at the above discussion, some conclusions can be drawn.

- An issue-based argument mediation system has the advantage that the process of argumentation has a focus, which can be useful, or even necessary (e.g., in a game-like situation). However, a system that allows free argumentation (such as the system presented in this paper), is more general and adds flexibility, since it not only allows justification, but also inference.
- Current argument mediation systems have different notions of defeasibility. It is as yet unclear which notions should be preferred. One should therefore strive for integration, or explicitly defend choices.
- A notion of rules should be included, or one should defend why it is not. Remarkably, none of the discussed systems with a visual, window-style interface has a notion of rules.

- Argument mediation systems with visual, window-style interfaces are obviously more user-friendly than text-based interfaces. Among the visual interfaces, a template-based interface seems easier to use than a graphical interface (as in the Argue!-system), in which special visual structures have to be drawn.

6 The relevance of computer-mediated defeasible argumentation

In my opinion, the following points make the development of implemented systems of computer-mediated defeasible argument worthwhile:

- Implemented systems for computer-mediated defeasible argumentation are *realizations* of formal models of defeasible argumentation. As a result, they provide an existence proof: they show that implementing the formal theory is feasible.
- Implemented systems for computer-mediated defeasible argumentation are *test beds* for formal models of defeasible argumentation. They are tools for experimentation, both technically and philosophically (which section 4 attempts to show).
- Implemented systems for computer-mediated defeasible argumentation can be *showcases*, giving formal models more credibility. There is a proviso here: the system's 'look and feel' must be good.
- Implemented systems for computer-mediated defeasible argumentation can be *practical aids*. Applications in law, decision making, planning and education are among the ambitions, which are even already partially met.

I think these points make the development of computer-mediated defeasible argument currently relevant for researchers (and not only for system developers). This holds not only for the first two more theoretically oriented points, but also for the second two more practically oriented ones. As yet, system developers alone cannot achieve the development of showcases of defeasible argumentation and practical aids. The early stage of development of computer-mediated defeasible argumentation necessitates a strong input from the research community.

Acknowledgments

The author gladly acknowledges the financial support by the Dutch National Programme Information Technology and Law (ITeR) for the research reported in this paper (project number 01437112). He also thanks Arno Lodder, Jaap Hage and Gerard Vreeswijk for comments and discussion.

References

Dung, P.M. (1993). On the acceptability of arguments and its fundamental role in nonmonotonic reasoning and logic programming. *IJCAI-93. Proceedings of the Thirteenth International Joint Conference on Artificial Intelligence* (ed. R. Bajcsy), pp. 852-857. Morgan Kaufmann Publishers, San Mateo (California).

Dung, P.M. (1995). On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and *n*-person games. *Artificial Intelligence*, Vol. 77, pp. 321-357.

Eemeren, F.H. van, Grootendorst, R., and Kruiger, T. (1981). *Argumentatietheorie.* Uitgeverij Het Spectrum, Utrecht.

Eemeren, F.H. van, Grootendorst, R. and Kruiger, T. (1987). *Handbook of Argumentation Theory. A Critical Survey of Classical Backgrounds and Modern Studies*. Foris Publications, Dordrecht. Translation of van Eemeren *et al.* (1981).

Gordon, T.F., and Karacapilidis, N. (1997). The Zeno Argumentation Framework. *The Sixth International Conference on Artificial Intelligence and Law. Proceedings of the Conference*, pp. 10-18. ACM, New York (New York).

Hage, J. (1996). A Theory of Legal Reasoning and a Logic to Match. *Artificial Intelligence and Law*, Vol. 4, pp. 199-273.

Hage, J. (1997). *Reasoning with Rules. An Essay on Legal Reasoning and Its Underlying Logic*. Kluwer Academic Publishers, Dordrecht.

Lin, F. (1993). An argument-based approach to nonmonotonic reasoning. *Computational Intelligence,* Vol. 9, No. 3, pp. 254-267.

Lodder, A.R. (1998). *DiaLaw – on legal justification and dialog games*. Dissertation. Universiteit Maastricht, Maastricht.

Loui, R.P. (1991). Ampliative Inference, Computation, and Dialectic. *Philosophy and AI. Essays at the Interface* (eds. Robert Cummins and John Pollock), pp. 141-155. The MIT Press, Cambridge (Massachusetts).

Loui, R.P. (1992). Process and Policy: Resource-Bounded Non-Demonstrative Reasoning. *Report WUCS-92-43*. Washington University, Department of Computer Science, Saint Louis (Missouri).

Loui, R.P., Norman, J., Altepeter, J., Pinkard, D., Craven, D., Lindsay, J., and Foltz, M. (1997). Progress on Room 5. A Testbed for Public Interactive Semi-Formal Legal Argumentation. *The Sixth International Conference on Artificial Intelligence and Law. Proceedings of the Conference*, pp. 207-214. ACM, New York (New York).

Nute, D. (1988). Defeasible reasoning: a philosophical analysis in Prolog. *Aspects of Artificial Intelligence* (ed. James H. Fetzer), pp. 251-288. Kluwer Academic Publishers, Dordrecht.

Pollock, J.L. (1987). Defeasible reasoning. *Cognitive Science*, Vol. 11, pp. 481-518.

Pollock, J.L. (1994). Justification and defeat. *Artificial Intelligence,* Vol. 67, pp. 377-407.

Pollock, J.L. (1995). *Cognitive Carpentry: A Blueprint for How to Build a Person.* The MIT Press, Cambridge (Massachusetts).

Prakken, H. (1993). *Logical tools for modelling legal argument.* Dissertation. Vrije Universiteit, Amsterdam.

Reiter, R. (1980). A Logic for Default Reasoning. *Artificial Intelligence,* Vol. 13, pp. 81-132.

Rittel, H.W.J., and Webber, M.M. (1973). Dilemmas in a general theory of planning. *Policy Sciences* 4.

Verheij, B. (1996a). Two approaches to dialectical argumentation: admissible sets and argumentation stages. *NAIC'96. Proceedings of the Eighth Dutch Conference on Artificial Intelligence* (eds. J.-J.Ch. Meyer and L.C. van der Gaag), pp. 357-368. Universiteit Utrecht, Utrecht. An abstract is available on the World-Wide Web at http://www.metajur.unimaas.nl/~bart/papers/cd96.htm.

Verheij, B. (1996b). *Rules, Reasons, Arguments. Formal studies of argumentation and defeat*. Dissertation. Universiteit Maastricht, Maastricht. A summary and table of contents are available on the World-Wide Web at http://www.metajur.unimaas.nl/~bart/proefschrift/.

Vreeswijk, G. (1993). *Studies in defeasible argumentation.* Dissertation, Vrije Universiteit, Amsterdam.

Vreeswijk, G. (1995). IACAS: an Implementation of Chisholm's Principles of Knowledge. *Dutch/German Workshop on Nonmonotonic Reasoning. Proceedings of the Second Workshop*, pp. 225-234. Delft University of Technology, Universiteit Utrecht.

Vreeswijk, G. (1997). Abstract argumentation systems. *Artificial Intelligence*, Vol. 90, pp. 225-279.