Argumentation support software: boxes-and-arrows and beyond

BART VERHEIJ†

Artificial Intelligence, University of Groningen, Groningen, The Netherlands

Presented at the workshop on 'Graphic and visual representations of evidence and inference in legal settings' at Cardozo School of Law, New York City, 28–29 January 2007.

This paper is about software to support argumentative tasks for lawyers. The focus is on the visualization and evaluation of warranted defeasible arguments. The opportunities provided by 'boxes-and-arrows' diagrams of arguments are treated, but the paper also addresses limitations. It is argued that finding natural design, improving usefulness and including content are central research challenges for the field of argumentation support software. It is of highest priority that claims about the extent to which these challenges have been met become better supported by evidence, e.g. by empirical studies.

Keywords: argumentation software; defeasible reasoning; legal logic.

1. Introduction

If Google is queried to define argumentation,[1] the first entry provided is as follows:

> a type of discourse in speech or writing that develops or debates a topic in a logical or persuasive way.

This is not the place to closely analyse the value, correctness and possible variations of this definition that Google took from the website of the Nebraska Department of Education (see Hitchcock, 2006, for a scholarly discussion of the concepts of argument and argumentation). This definition is reasonable and contains not much that would offend a specialist. For present purposes, however, it is worth noting that by its focus on speech and written texts this Google definition does not include the perspective on argumentation that underlies this paper: argumentation can be performed not only orally and verbally but also *visually*, i.e. in terms of diagrams, and *virtually*, i.e. using computer programs. In recent years, the visualization and virtualization of argumentation is receiving ever more attention. This paper is about the possibilities and the limitations of the visual and virtual approach to argumentation. The opinions concerning these matters presented here are grounded in my research into argument visualization, argumentation support software and the modelling of legal reasoning. Section 2 is about research into argument diagramming, argumentation software and defeasible logic with a focus on the ArguMed software and the logical system DefLog (Verheij, 2003a,b, 2005b). This work is one version of the boxes-and-arrows approach towards the visualization of

---

† Email: b.verheij@ai.rug.nl

[1] http://www.google.com/search?q=define%3Aargumentation. Page visited on 8 February 2007.

argumentation.[2] Section 3 deals with other elements of my work on legal reasoning. There the focus is on aspects of argumentation that go beyond boxes and arrows. Three research challenges are used as leitmotif of the paper: naturalness, usefulness and content. The paper concludes with a summary of these challenges that deserve significant attention of the field.

The paper has been written as output of the conference *Graphic and Visual Representations of Evidence and Inference in Legal Settings*, held at the Benjamin N. Cardozo School of Law in New York on 28–29 January 2007.[3]

1.1 *Argumentation software*

Argument assistants are computer programs that support argumentative tasks. In the more familiar setting of text writing tasks, supporting software is commonplace: text writing assistants are computer programs that support text writing tasks, and are known as word processing software. As yet, argument assistance software is evidently not nearly as successful as word processing software, which should not come as a surprise: most argument assistance software that has been built has not risen above the research prototype stage. Argument assistance software is a topic that still lies open for exploration and discovery. Things are changing, though, as we are seeing an increase of knowledge about the possibilities and limitations of argument assistants. This conference is a case in point here: as far as I know, there has not yet been such a high quality gathering of people building, thinking about and using argument assistants.

What I have to say here takes my own research into argument assistance software as a starting point, and in particular my system ArguMed (Verheij, 2003a, 2005b). A screen of ArguMed appears in Fig. 1. In the diagramming pane, one can see the topic at issue (indicated by a question mark), namely whether Peter shot George. There is also a witness testimony statement (According to witness A, Peter shot George) indicating a reason that could support that Peter shot George. In this case, the reason is, however, not successful in justifying that Peter shot George as there is an exception, viz. that the witness is unreliable.

ArguMed can be characterized by five points of focus.

- **Warranted defeasible argumentation**
  The arguments that can be built in ArguMed are defeasible, i.e. they do not always justify their conclusions. There can for instance be counterreasons and exceptions. The arguments in ArguMed can be warranted, in Toulmin's (1958) sense: it is possible to specify the generic inference license underlying a reasoning step. Somewhat more generally: ArguMed allows pros and cons reasoning and reasoning about whether a reason justifies its conclusion or defeats it. Section 2 provides further detail concerning the model of warranted defeasible argumentation used in ArguMed.

---

[2] Other examples of this approach include Van Gelder's Rationale software, Reed and Rowe's Araucaria system and Walker's visualization framework. See the work presented at the 2007 New York conference (note 3). My use of the phrase 'boxes-and-arrows approach towards the visualization of argumentation' is not intended to be clearly delimited. For instance, although several examples of the approach focus on tree structures (i.e. structures without cycles), this need not hold for all. For instance, the Argue! system that I developed before the ArguMed system allows cycles.

[3] The conference was excellently organized by Peter Tillers, Henry Prakken, Thomas D. Cobb and Jonathan Gottfried. Further details are available at http://tillers.net/conference.html (page visited on 8 February 2007).

ArguMed - peter shot george.def

File Edit Options

File | Theory | Dialectical arguments

? ○ *Peter shot George*

! ✚ Witness A is unreliable

! ✚ According to witness A, Peter shot George

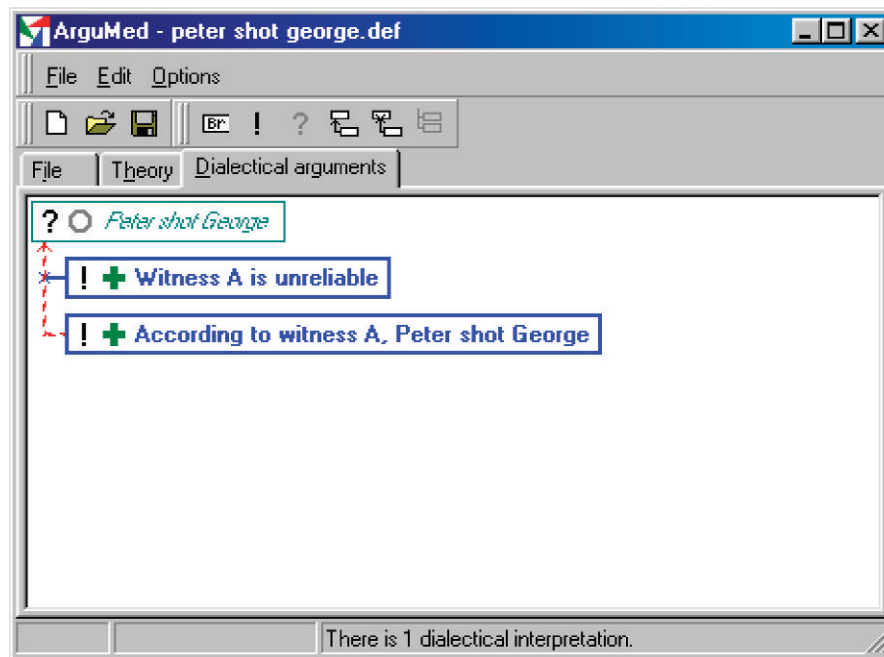There is 1 dialectical interpretation.

FIG. 1. A screen of ArguMed based on DefLog.

- **Argument construction, argument drafting**
  ArguMed was intended to support the construction and drafting of arguments as a dynamic process. This characterizes the difference between argument assistance software and automated reasoning systems. The latter are tools to perform reasoning tasks autonomously instead of providing support for humans performing such tasks. One reason to start investigating argument assistants is in fact that automated reasoning systems are of limited use in dynamic, open domains such as the law (see also Section 1.2).

- **Natural interaction, 'natural argumentation'**
  For argument assistants to be successful, they must allow interaction that feels natural for users. The steps that must be performed to fulfil an argumentative task should be easily recognizable for users. One way of trying to meet this requirement in ArguMed was to design a move-centred interface, instead of an interface based on graphical structures. The underlying idea was that whereas drawing graphical elements such as boxes and arrows directly is less natural than performing argumentative moves such as making a statement or providing a reason (cf. the buttons in Fig. 1). ArguMed has been qualitatively evaluated on the basis of a protocol-based user study (see Verheij, 2005b, for further information).

- **Argument evaluation**
  ArguMed is a tool that allows the construction of arguments, but also evaluates the arguments constructed. This is especially relevant when argumentative information can be added on the fly, and even more so in the case of defeasible argumentation: whether a conclusion is justified given

certain argumentative information can change when new information is added. For instance, new reasons against a conclusion can have the effect that the conclusion is no longer justified. See Section 2, in particular 2.4 onwards.

- **Underlying logic of defeasible argumentation**
  Argument evaluation as it occurs in ArguMed is logic based. In fact, ArguMed implements a formal logical system that models defeasible argumentation (DefLog; see Verheij, 2003b, for formal details). Initially, ArguMed was a system that implemented a logic of reasons with undercutters (in the sense of Pollock's undercutting defeaters; Pollock, 1987, 1995). In that original version, statements could be made and reasons could be provided for them. Reasons could be undercut, meaning that reasons could be given that blocked the justifying connection between the undercut reason and its conclusion. Later, the system grew into a logic of prima facie assumptions, which generalized the earlier undercutter logic. By the use of a conditional and a special kind of negation representing defeat, both support and attack could be expressed in the logic. Since the conditional allows nesting, it is also possible to express information about attack and defeat (see Section 2, especially 2.7, for further information about the logical system).

1.2 *A theory construction view on the law*

From the perspective of the specialized, interdisciplinary field of Artificial Intelligence and Law, research into argument assistance software can be regarded as a reaction to earlier attempts to fully automate legal reasoning.

In a legal setting, successful automated reasoning systems have been developed in domains in which the applicable regulations are very clear and complete about their conditions and the corresponding legal conclusions, and moreover in which the regulations are relatively stable. Clear and complete specification is a fundamental prerequisite for machine-understandable knowledge representation. The additional constraint of stability is a pragmatic one because even in cases where machine-understandable knowledge representation is possible, it is normally very costly to develop and maintain.

Especially, certain administrative tasks have been successfully automated (see, e.g. Oskamp and Lauritsen, 2002). But obviously many legal domains are not so clearly and completely specified, or
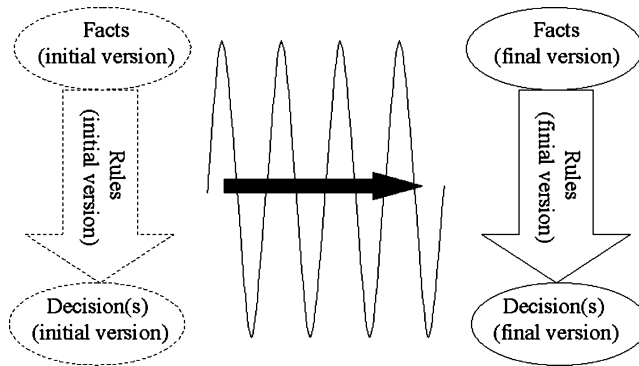

FIG. 2. A theory construction view on legal decision making (Verheij, 2003a, 2005b).

not sufficiently stable. As a result, several researchers have shifted their focus to reasoning support systems, of which argument assistance software is an example.

This shift from automated reasoning software to argument assistance software has an analogue in perspectives on legal decision making. One could say that the perspective underlying automated reasoning software is closer to the *subsumption model* of legal decision making, in which decisions follow from given facts and given rules. The subsumption model is related to Montesquieu's famous metaphor of the judge as a mere 'bouche de la loi'. The perspective on legal decision making underlying argument assistance software is fundamentally different. It is the *theory construction view* in which decision making consists of the gradual adaptation of one's initial theory of the facts, rules and decisions until one settles for a final theory. In this view, legal decision making is a process developing over time, limited by resource constraints, and driven by the critical assessment of the current theory of the facts, rules and decisions (cf. Fig. 2).

2. Modelling argumentation as boxes and arrows

In this section, I will summarize my work on the formal modelling of warranted defeasible argumentation, and the accompanying software tools in which arguments are presented in a boxes-and-arrows style. The presentation will be slightly different from my original work on the topic (Verheij, 2003a, 2005b). Here, I will take Toulmin's scheme as starting point as I did in 2005a (see also Hitchcock and Verheij, 2006), but with certain adaptations in the graphical representation that—hopefully—increase naturalness and recognizability.[4]

2.1 *Toulmin's argument model*

Toulmin (1958) introduced a model for the analysis of arguments that was richer than the traditional logical scheme focusing on premises and conclusions. His model has been and remains influential across a variety of disciplines (cf. Hitchcock and Verheij, 2006). The model is shown in Fig. 3. Datum and claim are analogues of premise and conclusion. Toulmin's original example used 'Harry was born in Bermuda' as datum and 'Harry is a British subject' as claim. The warrant is a generic inference license underlying the step from datum to claim. In the example: 'A man born in Bermuda will generally be a British subject'. The backing (the statutes and other legal provisions so-and-so obtain) provides support for the warrant. The rebuttal allows argumentation against the claim or the support for it (e.g. 'Harry has become a naturalized American'). Toulmin also included a qualifier, in order to make explicit that arguments can lead to a qualified conclusion (Presumably, Harry is a British subject).

2.2 *Datum and claim*

The basic, non-trivial form of argumentation consists of a reason that supports a conclusion. In Toulmin's terminology: a claim supported by a datum. In the present approach, in order for the claim to follow, two elements are needed: the datum and the connection between the datum and the claim. Figure 4 gives a graphical representation of the three basic situations. On top, there is the situation in which the datum and the connection between datum and claim are assumed

---

[4] More specifically, the use of exclamation and question marks to distinguish assumptions from issues was confusing, hence is dropped here. Also the arrangement of boxes and arrows has been adapted.

D ———————————➤ So, Q, C

Since
W

Unless
R

On account of
B

D for *Datum*   W for *Warrant*
Q for *Qualifier*   B for *Backing*
C for *Claim*   R for *Rebuttal*

FIG. 3. Toulmin's argument model.

| Harry was born in Bermuda | ➤ | Harry is a British subject |

| Harry was born in Bermuda | ➤ | Harry is a British subject |

| Harry was born in Bermuda | ·······➤ | Harry is a British subject |

FIG. 4. A datum and claim.

(indicated by the thick lines). As a result of these assumptions the claim is positively evaluated (indicated by the bold, green font).

Logically, this corresponds to a Modus ponens step:

$$\frac{D \qquad\qquad D \sim> C}{C}$$

Here, D stands for the datum and C for the claim. $D \sim> C$ is the conditional sentence that expresses the connection between the datum and the claim.

It can occur that the datum is considered a possible reason for the claim, while the datum itself is not assumed (see the middle of Fig. 4). In other words, the connection between datum and claim is assumed, but not the datum itself. Then the datum and claim are each neither positively nor negatively evaluated, which is indicated by a regular, black font and a dotted border. When the datum is not assumed, argumentation can naturally proceed by providing a reason for the datum (turning the datum into the claim of a second datum/claim pair).

The bottom of Fig. 4 shows the situation where the datum is assumed, but the connection between datum and claim is not. In that case, the datum is positively evaluated, but the corresponding claim is not.

In logical terms, this analysis can be summarized like this. If both D and D $\sim>$ C are assumed as premises, C follows. Conversely, if D or D $\sim>$ C (or both) are missing, C does not follow (i.e. C does not follow from this datum; there could be another datum from which the claim follows).

2.3 *Warrant and backing*

When the conditional connection between datum and claim is not assumed (as at the bottom of Fig. 4), a natural argumentative move is to specify the warrant that gives rise to it. A warrant is a generalized inference license and can as such in ordinary language be phrased as a rule sentence. In this connection, it is important to distinguish between the following three.

A man born in Bermuda will generally be a British subject.

If *Person* was born in Bermuda, then generally *Person* is a British subject.

If Harry was born in Bermuda, then generally he is a British subject.

The first is the warrant, i.e. a generic inference license phrased as a rule sentence. The second is the scheme of specific inference licenses related to it. It is phrased as a conditional sentence with a variable (*Person*). The third is a specific inference license. It is one of the instances of the scheme preceding it. The first two (the warrant and the associated scheme) are in a specific sense equivalent since either expresses the warrant's core meaning as a generic inference license. However, obviously only the rule phrase occurs in ordinary language.

Figure 5 (top) shows a warrant that is assumed to hold and hence is positively evaluated. As a result of the warrant, the connection between the original datum and claim follows. Logically, this can be analysed as two chained instances of Modus ponens:

$$\frac{\dfrac{W \qquad\qquad W \sim> (D \sim> C)}{D \qquad\qquad D \sim> C}}{C}$$

It should be noted that, in this analysis, the generic nature of the warrant is not made explicit in the logical formula. Instead, the analysis stresses that the warrant, which by its nature is generic, implies a specific inference license (by the use of the nested conditional $W \sim> (D \sim> C)$). The reason for this choice of analytic style is the propositional nature of the logical formalism (as opposed to a style using predicates, terms and variables, which would allow a more direct representation of the genericness of warrants). Using a propositional style allows the formalism to be especially close to the graphical representation: elementary propositions are boxes and conditional propositions correspond to arrows. In fact, it is a key feature of ArguMed and its underlying logic DefLog that the arrows in the graphical representation are logically analysed as conditionals.

When the warrant is not assumed (Fig. 5, bottom), the connection between datum and claim does not follow. Consequently, the claim does not follow either and is not positively evaluated.

In such a case, a backing from which the warrant follows can be given as support for the warrant. The result is that the claim becomes positively evaluated again (Fig. 6).
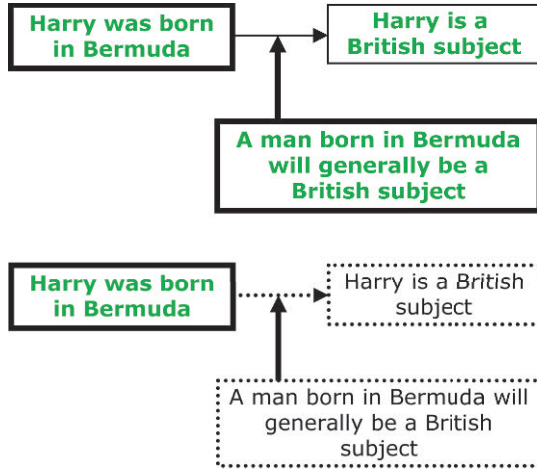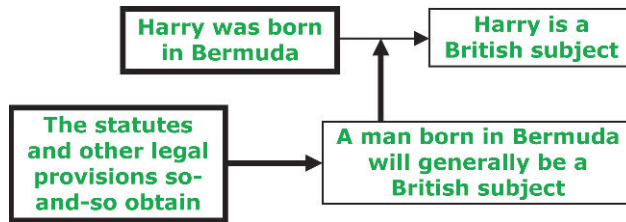
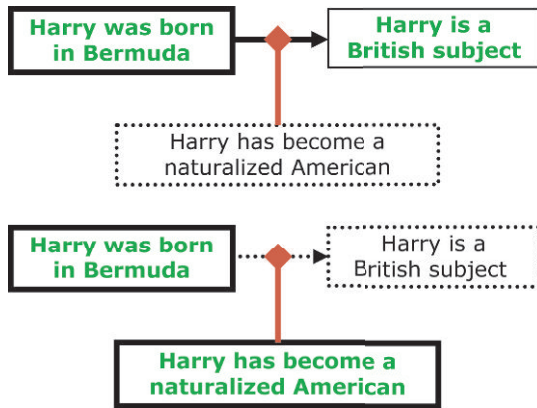FIG. 5. Adding a warrant.

FIG. 6. The role of a backing.

FIG. 7. A rebuttal (no warrant).

2.4 *Rebuttal I (no warrants)*

The strongest deviation from classical logical analyses of argument in Toulmin's model is the notion of rebuttal. The possibility of rebuttals involves the defeasibility of arguments: an argument that shows why a conclusion follows can by new information (counterreasons, exceptions to a rule, etc.) become overturned. Technically, the effect is non-monotonicity, i.e. it can occur that conclusions that initially follow are retracted given additional information.

The basic form of rebuttal and its effect on argument evaluation is most easily illustrated in a situation with only a datum and a claim, and no warrant or backing. Figure 7 shows the effect a rebuttal that attacks the connection between datum and claim. At the top, the rebuttal is not assumed (but its rebutting effect is assumed, as is shown by the thick red arrow with a diamond end); hence, the claim is still positively evaluated. At the bottom, the rebuttal is assumed, and therefore blocks the connection between datum and claim. As a result, the claim is no longer positively evaluated (see also Fig. 1).

2.5 *Reinstatement*

An important phenomenon that can occur when argumentation is defeasible is reinstatement. This occurs when a conclusion follows, then by additional information no longer follows, but subsequently—when even more information is added—follows again. For instance, assume an unlawful act case in which someone has broken a window. As a result, it can at first be argued that he has an obligation to pay for the damages because of breaking the window. That conclusion no longer follows when this argument is attacked by a ground of justification, e.g. because the breaking of the window allowed the saving of a child in the burning house. The obligation to pay can become reinstated again in case it turns out that the original aim of entry was theft. Figure 8 shows the end point of this exchange of reasons and counterreasons.

As a side issue, the example can be used to illustrate the need for 'rebuttal warrants'. In the example, it is debatable whether (in an actual legal system) an original, unlawful aim of entry (here theft) blocks the inference that saving the child gives rise to a ground of justification. It therefore can be relevant to specify the underlying generic rebuttal license (something like 'When the aim of entry is unlawful, force majeure does not give rise to a ground of justification'), and give backing for that. Such rebuttal warrants cannot be treated in Toulmin's analysis, as that allows only 'support warrants'. In the present analysis and in the ArguMed tool, rebuttal warrants are available.
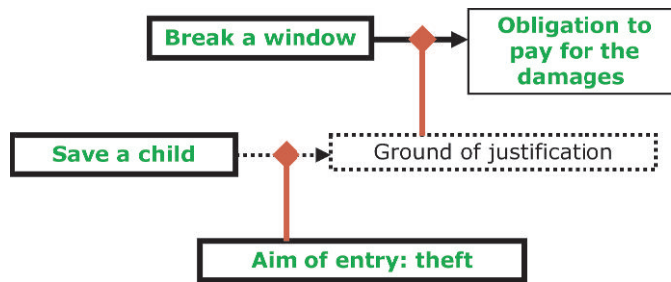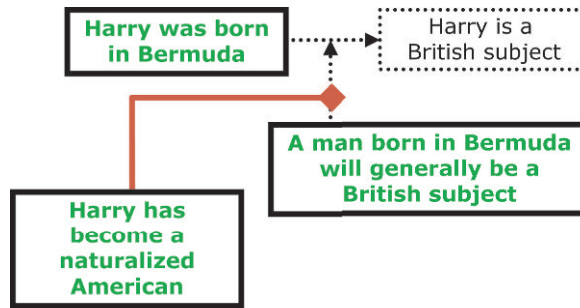

FIG. 8. Reinstatement.

FIG. 9. A rebuttal (in the presence of a warrant).

2.6 *Rebuttal II (with warrants)*

If we look at the warrant–datum–claim part of Toulmin's scheme (cf. Fig. 5), there are five statements that can be argued against:

- The datum D.

- The claim C.

- The warrant W.

- The implicit conditional 'If D, then C' that expresses the bridge from datum to claim.

- The implicit conditional 'If W, then if D, then C' that expresses the bridge between warrant and the previous implicit conditional.

An especially relevant kind of attack is that of blocking that a warrant is instantiated. In other words, in such an attack, it is blocked that a warrant gives rise to the relevant, specific inference license (Fig. 9). This corresponds to an attack of the fifth statement in the above list.

2.7 *DefLog*

The evaluation mechanism illustrated above has an underlying logical system called DefLog (Verheij, 2003b). It uses a simple logical language with two propositional connectives, one expressing a conditional relation, the other expressing a specific kind of negation.

The conditional, denoted by $\sim>$, only validates Modus ponens. There is no introduction rule (as there is for the standard truth-functional conditional). The idea is that the conditional just represents a specific inference license, and nothing more. Moreover, the expressed inference licenses are intended to be concrete and substantial, and hence are a 'matter of the premises'. One could say that the conditional expresses primitive, substantial implication.

The negation, denoted by $\times$, expresses the defeat of a prima facie assumption. In combination with the conditional it can express attack: a sentence R $\sim>$ $\times$C expresses that R attacks C.

Verheij (2003b) defines a semantics for this language. It is a variant of the stable semantics as they commonly appear in the context of non-monotonic logics. The core definition is that of a dialectical interpretation of a set of sentences Δ. It is included here for the purpose of illustration. The elements of the set Δ are the prima facie assumptions. The set is divided into two disjoint subsets J and D. The elements of the former are the justified assumptions, the latter the defeated assumptions. By

definition, such a partition (J, D) is called a *dialectical interpretation* of Δ if two conditions are fulfilled:

1. J is conflict free, i.e. there is no sentence S in J of which the defeat sentence ×S is in J or follows from J (using Modus ponens).
2. The defeat sentence ×S of each element S of D is in J or follows from J (using Modus ponens).

If (J, D) is a dialectical interpretation of a set of prima facie assumptions, each consequence of J (using Modus ponens) is justified.

It should be noted that, as a consequence of this definition, it need not be the case that all prima facie assumptions are actually justified; some can be defeated by the justified assumptions. The defeated assumptions are the elements in the set D of a dialectical interpretation (J, D).

It turns out that a set of prima facie assumptions can have more than one dialectical interpretation (a kind of ambiguity) or has none (a kind of inconsistency). This is characteristic for stable semantics. DefLog also has an analogue of Dung's (1995) preferred semantics, which lacks inconsistency, in the sense that all sets of prima facie assumptions have an interpretation in the preferred semantics (see Verheij's, 2003b, discussion of dialectical justification for details).

DefLog extends Dung's (1995) abstract argumentation systems, as formally shown by Verheij (2003b). An argumentation framework in Dung's sense can trivially be translated to a DefLog theory: when an argument A attacks an argument B, this is expressed as A ∼> ×B. DefLog's language is more expressive than Dung's. By adding a conditional, support can be explicitly expressed, and since both support and attack are expressed in the logical language, it is possible to model argumentation about support and attack, as exemplified by the present treatment of warrants and undercutters.

Whereas Pollock (1987, 1995) uses two primitives to express defeat (undercutting and rebutting defeaters), in DefLog one primitive suffices (the dialectical negation). For instance, when U is an undercutting defeater (in Pollock's sense), that blocks the connection between reason R and conclusion C, then U's undercutting effect is represented in DefLog as follows:

$$U \sim> \times(R \sim> C).$$

In words: given U, it is defeated that C can be inferred from R. Rebutting defeat involves a reason R1 for the conclusion C, and a reason R2 against the conclusion C. If R1 can defeat R2 (R1 is 'stronger' than R2, or R1 'outweighs' R2), then R1 is a rebutting defeater. Using DefLog's primitives, the rebutting effect can be represented as follows:

$$((R1 \sim> C) \wedge R1) \sim> \times(R2 \sim> \neg C).$$

Here, the standard connectives $\wedge$ for conjunction and $\neg$ for negation are used to extend DefLog's proper expressiveness. In words: it is defeated that the negation of C can be inferred from R2 when R1 successfully implies C, i.e. when R1 obtains and it also holds that C can be inferred from R1.

Next to these two important kinds of defeat, it is in DefLog possible to express other kinds. For instance, it is possible to express defeat relations between elementary propositions: when one prima facie assumption attacks another, this can be expressed as A1 ∼> ×A2. Another relevant defeating effect (related but not equal to Pollock's rebutting defeat) is the following:

$$C \sim> \times(R \sim> \neg C).$$

In words: if a claim already obtains, its negation cannot successfully follow from a reason against it. This kind of defeating effect is closely related to the way in which default rules in Reiter's (1980, 1987) logic for default reasoning can become blocked.

DefLog has been implemented in the software tool ArguMed (available at http://www.ai.rug.nl/ ∼verheij/aaa/). Conditional sentences are graphically represented using boxes connected by arrows (see Fig. 1). The evaluation mechanism in ArguMed computes the stable semantics.

3. Beyond boxes and arrows

Notwithstanding the fact that I strongly value and support the developments concerning argument assistance software and the associated increasing knowledge about argument diagramming, of which Section 2 is only one example, it is also true that my work has increased my awareness of the limitations of such software and diagramming. In this section, I will try to articulate why in my opinion it is also necessary to continue investigations that go beyond boxes and arrows. The following subsections about Reason-Based Logic, argumentation schemes, legal skills and ArguGuide will each illustrate aspects of legal reasoning that are not easily accommodated in the boxes-and-arrows approach.

3.1   *Reason-Based Logic*

Reason-Based Logic (Hage, 1996, 1997; Verheij, 1996) is a logical formalism in which reasoning with rules and the reasons based on them is modelled. Its design is guided by the way in which rules and reasons are used in the field of law. Rules are treated as objects with properties. For instance, the validity of a rule is expressed using a formula of the following form:

$$\text{Valid(rule(}\textit{condition}, \textit{conclusion}\text{)).}$$

Here, *condition* and *conclusion* are variables that stand for the condition and conclusion of a rule. 'Valid' is a special predicate symbol of the language of Reason-Based Logic and 'rule' a special function symbol. A rule is generic in the sense that its condition and conclusion have a range of instances. An example is the following formal version of the rule that thieves are punishable:

$$\text{Valid(rule(thief(}\textit{person}\text{), punishable(}\textit{person}\text{)).}$$

The rule condition thief(*person*) has instances thief(john), thief(mary) where the logical terms john and mary stand for concrete persons. The specific use of uppercase and lowercase characters is part of Reason-Based Logic (see the mentioned original sources for details on this).

When a rule is applied, its conclusion does not directly follow, but instead first results in a reason for it. For instance, when the rule that thieves are punishable is applied, we can get the following:

$$\text{Reason(thief(john), punishable(john)).}$$

When a rule's condition is fulfilled, i.e. in the example, when John is a thief, this normally leads to a reason for the corresponding instance of the rule's conclusion. However, not always: the applicability of the rule can be excluded, and then no reason follows. The exclusion of the rule is expressed thus:

$$\text{Excluded(}\textit{rule}\text{, thief(john), punishable(john)).}$$

Here, *rule* stands for the rule that thieves are punishable. The formula expresses that the applicability of the rule is excluded. Note that a rule's exclusion is linked to an instance of the rule: here,

the rule's applicability is only excluded for its instance for John. For other instances, the rule's applicability need not be excluded. The reason for this design choice is that exclusionary reasons (i.e. the reasons for the exclusion of a rule) occur in a concrete case, and do not exclude applicability of the rule in general.

When there is a reason, normally the conclusion supported by the reason follows. However, a central element of Reason-Based Logic is its modelling of the weighing of reasons. When there is a conflict of reasons, i.e. when there is both a reason for a conclusion and one against it, the conclusion only follows when the pros outweigh the cons:

$$\text{Outweighs}(\textit{reasons\_pro}, \textit{reasons\_con}, \text{punishable(john)}).$$

Here, *reasons_pro* and *reasons_con* stand for the reasons for and the reasons against John being punishable, respectively. In Reason-Based Logic, the result of the weighing of reasons is not calculated (e.g. by counting the numbers of reasons or on the basis of numerical values expressing reason strength), but is instead explicitly represented. In other words, which reasons outweigh which other reasons is a form of knowledge about a domain. One relevant logical property related to the weighing of reasons is encoded in Reason-Based Logic, namely the property that when a set of pros outweighs a set of cons, adding more pros does not change the result of weighing.

Hage (1996, 1997) and Verheij (1996) give much further information about Reason-Based Logic, including examples and connections with issues in philosophy, logic and the law. For instance, Reason-Based Logic has been applied to the discussion about rules and principles in legal theory. Dworkin (1978) claimed that legal rules and legal principles are to be *logically* distinguished. He argued that whereas rules lead to their conclusions directly, principles give rise to reasons only. Similarly, conflicting rules lead to logical contradictions, whereas conflicting principles do not. Principles lead to reasons, which can be weighed in case of conflict. An analysis in Reason-Based Logic (Verheij *et al.*, 1998) can be used to show that the distinction between rules and principles is not necessarily a logical distinction, as Dworkin claims. It turns out that it is possible to give an integrated logical model that does justice to the different behaviour of rules and principles as mentioned. In the analysis, rules and principles are the extremes of a range of possibilities of hybrids between rules and principles.

Can reasoning with rules and reasons as modelled in Reason-Based Logic be supported by argumentation software tools, such as ArguMed? This question does not allow a simple yes or no answer, so let us discuss some of the characteristic elements of Reason-Based Logic one by one.

- **Rules and reasons**
  The relation between datum, claim and warrant is closely related to the way in which in Reason-Based Logic a rule gives rise to a reason for its conclusion. There is an important difference in expressiveness, though, namely the representation of genericity. In Reason-Based Logic, such genericity can be expressed using variables and their instances, but in boxes-and-arrows diagrams (in the style used here) it is not clear how this should be done. In this connection, it is useful to recall the discussion in Section 2.3 about the logical modelling of generic inference licenses, the corresponding schemes of specific inference licenses and the specific inference licenses themselves. Walker's visualization approach (also presented at the 2007 New York conference; cf. Walker, 2007) provides an interesting attempt to integrate variables and their instances in boxes-and-arrows diagrams.
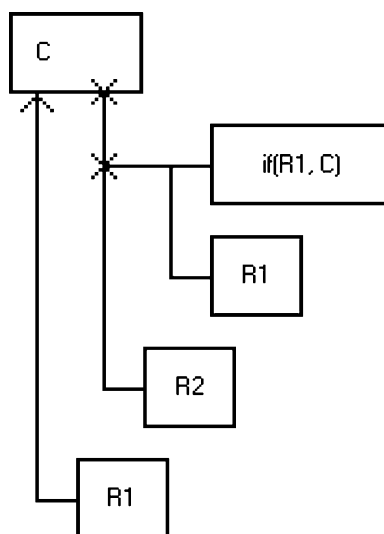
F<small>IG</small>. 10. Graphical representation of a rebutter and its defeating effect.

- **Exclusionary reasons and the weighing of reasons**
  Exclusionary reasons are closely related to one of the forms of rebuttal in the presence of a war-
  rant as discussed in Section 2.6, namely that of blocking the instantiation of a warrant (as shown
  in Fig. 9). Although the ArguMed tool can straightforwardly represent a collection of pros and
  cons concerning a particular issue, the weighing of reasons cannot be directly represented in it.
  Still, it is possible to represent the defeating effect of weighing as a sentence in the logical lan-
  guage of DefLog (in fact as a generalization of the representation of the defeating effect of a
  rebutting defeater in Pollock's sense, as discussed in Section 2.7). Moreover, since ArguMed is
  based on DefLog, such a sentence allows for a graphical representation. This representation is
  tied to the specific design of ArguMed and DefLog (Fig. 10). The figure shows the two reasons
  R1 and R2 for and against C. It also shows that R2 does not lead to C given both R1 and
  the conditional if R1, then C. In the specific context of ArguMed and DefLog, the representation
  in the figure is understandable and principled. The trouble is that the nature of that representation
  is not very natural or useful in practice. It is a matter of future research whether and, if so, how
  the weighing of reasons can naturally and usefully be supported and represented in argumentation
  software.

- **Rules and their properties**
  A central design choice underlying Reason-Based Logic is that rules are structured objects with
  properties. A central rule property made explicit in Reason-Based Logic is rule validity. As dis-
  cussed just now when addressing rules and reasons, ArguMed and DefLog allow for a limited
  representation of rule validity. But, in Reason-Based Logic, it is explicitly allowed to express
  other properties of rules than rule validity as well. This can be done using dedicated, domain-
  specific predicates. For instance, in the domain of law, a debate on a legal issue can involve
  the purpose of a regulation. Since a boxes-and-arrows representation uses propositions as basic

building blocks (the boxes) and conditionals as connections (the arrows), there is no obvious place for the representation of rule properties. Whether the representation of rule properties can be naturally and usefully supported in argumentation tools is a relevant direction for future research.

In sum, Reason-Based Logic's expressiveness can to some extent be accommodated within boxes-and-arrows diagrams, but goes in relevant respects beyond such representations. In particular, it is unclear whether, and if so, how the weighing of reasons and rule properties can be naturally and usefully dealt with. There are several options:

- Including the expressiveness of Reason-Based Logic precludes natural and useful argumentation software of the boxes-and-arrows style.

- Reason-Based Logic (better: the features of legal reasoning that it tries to capture) can be accommodated in boxes-and-arrows, but we have to further invent and explore how.

- Reason-Based Logic can be included in natural and useful argumentation software, but this requires an adaptation of or deviation from boxes-and-arrows diagramming.

3.2 *Argumentation schemes*

Argumentation schemes can be regarded as a generalization of the rules of inference of formal logic. The notion of argumentation schemes stems from the field of argumentation theory. Walton's (1996) discussion has been influential and is a useful overview with many examples. There also further references to the argumentation theory literature can be found. Here are two informal versions of logical rules of inferences.

1. P. If P then Q.
   Therefore Q.
2. All Ps are Qs. Some R is not a Q.
   Therefore some R is not a P.
   The former is an informal version of Modus ponens, the latter is one of the classically studied syllogisms (the one with mnemonic 'baroco'; cf. Chambers, 1728). These two schemes fit in neat formal systems; the former in many logical proof systems, but in particular in standard propositional logic, the latter in the classification of syllogisms. Both are truth preserving (in the sense that the truth of the schemes' conditions are taken to guarantee the truth of their conclusion) and allow no exceptions. Contrast these with the following two schemes.
3. Person E says that P. Person E is an expert with respect to the fact that P.
   Therefore P.
4. Doing act A contributes to goal G. Person P has goal G.
   Therefore person P should do act A.

The former is a variant of argumentation from expert opinion, the latter a variant of means-end reasoning. Although these schemes are recognizable as patterns that can occur in actual reasoning and are also—to some extent—reasonable, it is immediately clear that the neatness and safeness of the schemes (1) and (2) does not apply to (3) and (4). There is no clean formal system associated with (3) and (4), nor is one to be expected. They are not truth preserving. For (3), it suffices to note that an expert can be wrong and, for (4), it is even unclear how to establish the truth of its conclusion.

Furthermore, (3) and (4) allow exceptions. For instance, an exception to scheme (3) occurs when an expert has a personal interest in saying that *P*. For scheme (4), it can be the case that there are other, better ways to achieve goal *G*, or it may be impossible to do *A*.

Argumentation schemes are context dependent, defeasible and concrete instead of universal, strict and abstract. To some, these properties may seem to make argumentation schemes a useless tool of analysis, but it turns out that the properties are in fact what makes argumentation schemes useful. At the same time, argumentation schemes require a different way of approaching the analysis of reasoning than one in terms of neat logical systems. And although a full formalistic approach is not feasible given the nature of argumentation schemes, it is not necessary to proceed without any systematicity. In Verheij (2003c), I have defended to take inspiration from knowledge engineering practice, i.e. the process of extracting knowledge from domain experts and representing that knowledge in such a way that a machine can process it. I proposed the following four-step method for the investigation of argumentation schemes.

1. Determine the relevant types of sentences.
2. Determine the argumentation schemes.
3. Determine the arguments against the use of the argumentation schemes.
4. Determine the conditions for the use of the argumentation schemes.

Can argumentation schemes be naturally and usefully be supported in argumentation software? As yet, there is limited knowledge in this respect. The ArguMed system does not include argumentation schemes support (although there is some very preliminary work in this direction). Especially, the Araucaria system (see, e.g. Reed and Rowe, 2005) provides relevant first steps in this direction. In line with the focus of Araucaria, support is provided for the analysis of argumentative texts in terms of given argumentation schemes.

Concerning the inclusion of argumentation schemes in argumentation software, further work is required, especially concerning usefulness and naturalness. Also the question whether it is possible to provide support outside argument analysis deserves future attention. It would be especially interesting to investigate whether argument production on the basis of given argumentation schemes can be usefully supported. Interestingly, the latter would imply a step back towards automated reasoning tools, whereas one reason for the development of argumentation software was the wish to overcome the limitations of such tools (cf. Sections 1.1 and 1.2).

3.3   *Legal skills: writing case solutions*

With my former colleagues at the Faculty of Law at the University of Maastricht, I have developed course material for the training of argumentative legal skills (Verheij *et al.*, 2004). The focus was on solving legal cases, i.e. determining the legal consequences of a given legal case, and on writing an argumentative text in a structured and legally correct way.

Two elements are central in the underlying approach: rule analysis and condition evaluation. Rule analysis concerns the determination of the relevant parts of law. Rules are analysed in terms of their condition–conclusion structure. Rule conditions can have cumulative and alternative subconditions. Moreover, a rule analysis involves a search for alternative rules for the same or opposite conclusion and for rules with a (sub)condition or its opposite as their conclusion. Statutory law and authoritative precedents are used as the main sources of rules. Figure 11 provides an example in Dutch of (a graphical representation of) such a rule analysis.

Art. 3:44 lid 1 BW

Rechtsgevolg:
    De rechtshandeling is
    vernietigbaar

Voorwaarde a:
    De rechtshandeling is
    door bedreiging tot
    stand gekomen

Voorwaarde b:
    De rechtshandeling is
    door bedrog tot
    stand gekomen

Voorwaarde c:
    De rechtshandeling is
    door misbruik van
    omstandigheden tot
    stand gekomen

Nadere analyse

(Het rechtsgevolg
komt overeen met
de voorwaarde)

Art. 3:44 lid 2 BW

Rechtsgevolg:
    Bedreiging is aanwezig

Voorwaarde 1:
    Iemand beweegt een
    ander tot het verrichten
    van de rechtshandeling
    door deze of een derde
    met enig nadeel in
    persoon of goed te
    bedreigen

Voorwaarde 2:
    De (in voorwaarde 1
    genoemde) bedreiging
    is onrechtmatig

Voorwaarde 3:
    Een redelijk oordelend
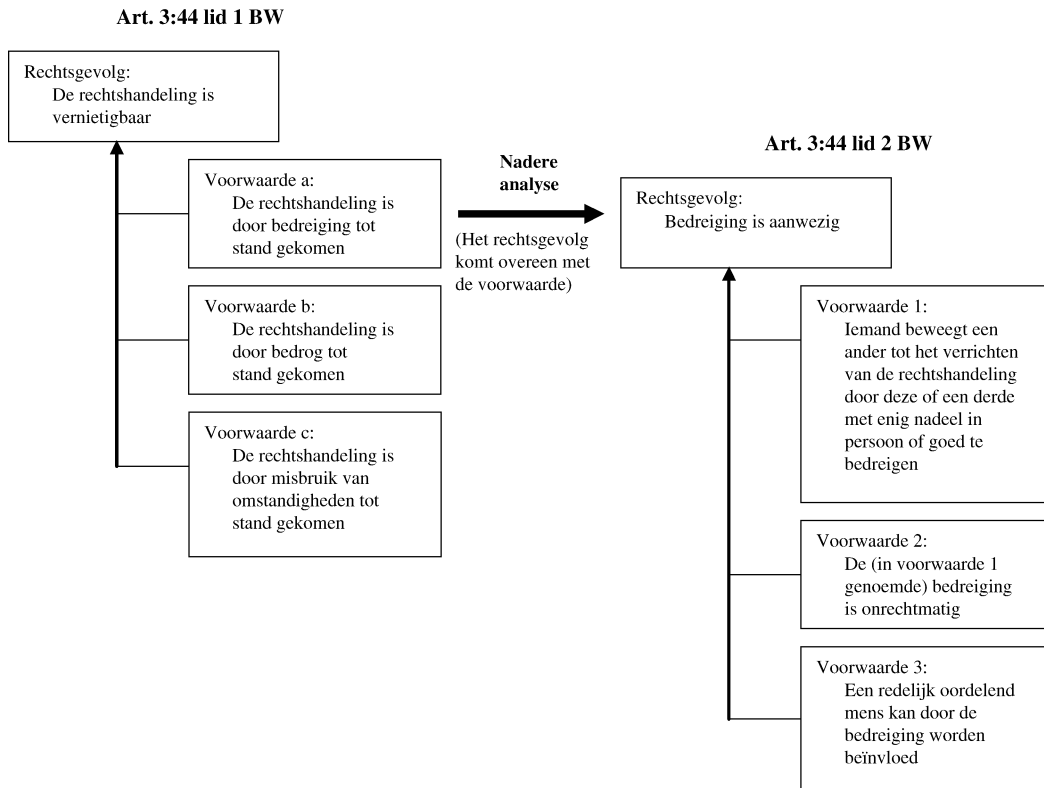   mens kan door de
    bedreiging worden
    beïnvloed

FIG. 11. An example of rule analysis (taken from Verheij *et al.*, 2004).

Condition evaluation consists of applying the law to the case at hand. A condition of an analysed rule is evaluated either by specifying the facts of the case that match the condition (or its opposite) or by referring to one or more other rules with the condition (or its opposite) as its conclusion.

In an early version of the material, students had to perform the rule analysis and the associated condition evaluation systematically and explicitly. This involved a kind of bookkeeping in which the logical relations between the rules, their elements and the case facts were encoded. Because of the tediousness of this process, it turned out that it was most productive to skip the systematic explicit approach quickly in order to keep the attention of students, and focus instead on the actual goal: writing good legal argumentative texts.

This was in fact in line with an experience while writing the course material: it was not at all easy to find meaningful and instructive examples that really fitted the systematic bookkeeping approach. Among the lessons learnt from this work (which involved around 400 students and approximately 8 teachers yearly) are the following.[5]

---

[5] It should be noted that these lessons are based on extensive teaching experience, but that no formal empirical experiments have been performed.

- **Provide examples of written case solutions, instead of a theoretical framework**
  It turned out that good examples of written case solutions were much more easily followed by students than the systematic bookkeeping approach itself. The systematic approach served more as a way of showing the key elements and of the necessity for precision than as an approach that needed to be performed perfectly.

- **Focus on examples with meaningful content, not on formal ones**
  We did not teach any formal logic. No Modus ponens, not even the fallacy of affirming the consequent. Although there might be some use (for instance in the context of teaching fallacies, an appropriate topic that we, however, did not address), there was no obvious need for formal examples for the actual solving of legal cases and writing about them. It turned out again and again that formal examples led to more confusion (both in students and in teachers) than to useful lessons learnt.

- **Train an analytic attitude, not a particular formal system**
  Before using the material in the book that we developed (Verheij *et al.*, 2004) the emphasis in the training and in the examination was more on adherence to the bookkeeping system and less on the real goal: writing good case solutions. The hope was that mastering the (semi)formal system associated with the bookkeeping system would easily be followed by the ability to write good case solutions. This hope was futile. It turned out to be much more effective to train an analytic attitude towards the written case solutions themselves. Pointing out the good and the bad elements in the case solutions (and associating an explicit grading system with that) improved teaching results significantly. Interestingly, from an abstract, 'deep structure' point of view, the relevant elements in the evaluation exactly matched the required elements of the bookkeeping system. But the text-oriented, instead of bookkeeping-oriented approach proved to be more effective as a teaching tool.

- **Show content-based difficulties in examples**
  A good reason for more emphasis on the training of a logically oriented bookkeeping approach towards writing case solutions would be that it is the logical structure that is hard. But it turned out that this is never (better: almost never) the case. Solving legal cases is hard because of its content, not because of its logical structure. Students did not make many relevant logical mistakes, for instance while identifying whether certain rule conditions are cumulatively necessary for a legal consequence or whether they are alternatives. Sometimes they do, but only if the legal topic has a very technical instead of a concrete meaning and if at the same time the wording of the associated legal source is obscure. These are cumulative conditions: if the wording is clear, a technical meaning does not lead to a logical problem; if the meaning is concrete, an obscure wording does not lead to a logical problem. A much more important source of errors is the legal content itself: which conditions must be fulfilled? Which rules apply? Which precedents are relevant and why? Which facts make that a condition is fulfilled or not? etc. Unfortunately, answers to questions like these are not given by a logically oriented bookkeeping method, but require the close study of source material and feedback from knowledgeable teachers.

These lessons concerning legal skills have a connection with the topic of this paper, namely the possibilities and limitations of argumentation software: they suggest that argumentation support in the domain of law should pay due attention to content, and not, or not predominantly, to logical structure. This idea has been taken up in the work reported on in Section 3.4.

3.4   *ArguGuide: argumentation support in terms of the knowledge structure of a legal topic*

Today's argumentation software lays much emphasis on the logical structure of reasoning, and then especially on the structure as it can be represented in boxes-and-arrows style diagrams. The effectiveness of this type of argumentation software is as yet empirically underpinned to only a limited extent (cf. van den Braak *et al.*, 2006; see Verheij, 2005b, for information on the user evaluation of ArguMed). Acquiring more knowledge about this is a significant topic of future research. At the same time, it seems to be worthwhile to investigate other ways of providing argumentation support, ways that are alternative or complementary to the diagrammatic argument structuring. The experiences concerning legal skills training reported upon in Section 3.3 provide a case in point here. These suggest that content-oriented argumentation support might be a sensible addition to diagramming methods. For instance, it may be more true than we (as researchers committed to argumentation support software based on diagrams) want that a legal professional is not helped very much with visualizations. It might be that checklists for legal content, memory extensions (for instance to keep large case records accessible) and integrated software for word, content and argument processing are more effective supplementary support tools in legal practice.

Following this suggestion, we have started research on argumentation support software that takes advantage of the knowledge structure of a legal topic. In her master project, Maaike Schweers has designed, built and evaluated the system ArguGuide, of which a screenshot is available in Fig. 12. The left pane shows the knowledge structure of the legal topic. Here, the key elements of cases concerning unlawful acts and the associated damage compensating obligations (as they pertain to Dutch civil law) are listed. The elements are listed in a way that resembles the hierarchical structure
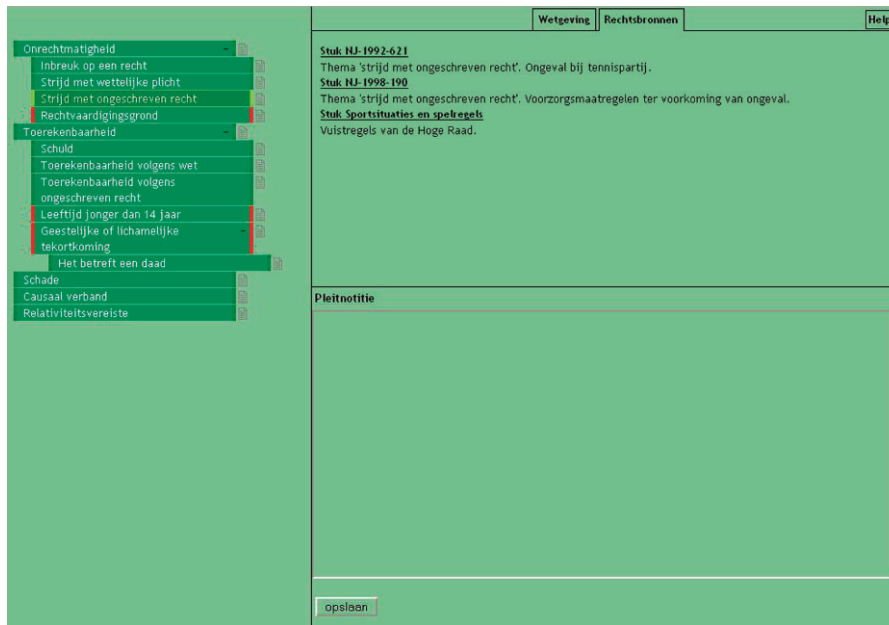

FIG. 12. A screenshot of ArguGuide, a support tool using the knowledge structure of a legal topic.

of the table of contents of a handbook on tort law. Some logical structure is retained. For instance, the four relevant subconditions determining unlawfulness (Onrechtmatigheid) are listed as subelements. However, the fact that the first three ('Inbreuk op een recht', violation of a right; 'Strijd met wettelijke plicht', violation of a statutory duty; 'Strijd met ongeschreven recht', violation of unwritten law) are alternative conditions for unlawfulness is not represented, which does not seem to be an omission for the system's intended users. The fourth element ('Rechtvaardigingsgrond', ground of justification) is an exception precluding unlawfulness. This is graphically indicated by showing a small red edge on either side of the element. In fact, explicitly showing this logical aspect was a design requirement based on expert interviews: experts asked for support related to burden of proof.

Each knowledge element has an associated box (accessible by clicking the icon on the right of each element) to type notes related to the element. These note boxes are provided to facilitate the drafting of argumentative texts. Such draft notes could be based on the legal content that can be investigated in the pane on the top right. The pane provides access to legal sources (statutory law or other legal sources) that are relevant for the knowledge element selected in the knowledge structure pane on the left. In the figure, the element 'Strijd met ongeschreven recht' is selected giving access to certain selected relevant decisions by the Dutch Supreme Court. The bottom right pane (Pleitnotitie) is an area in which an argumentative text can be written, possibly based on the draft notes produced in the knowledge pane.

The system design was based on predetermined requirements concerning functionality, flexibility, freedom and user friendliness. A qualitative user evaluation has been performed using a small number of test persons. The test persons were asked to work on a draft of an argumentative text concerning a particular legal case. The evaluation showed that the system was largely considered successful in terms of the design requirements. Generally, a positive attitude towards the system was shown, although it was doubted whether it would be an effective tool for an experienced lawyer. The latter is not surprising given the simple nature of the represented knowledge and of the information provided (only a very limited number of legal sources were made available). Test persons mentioned education as a possible application domain for the tool. On the whole, the direction chosen in ArguGuide appears to be a fruitful starting point for further attempts to build a content-oriented argumentation support tool.

4. Challenges

Above, it has been attempted to illustrate the possibilities of argumentation support software with a focus on the domain of law. Also a number of directions of future research and possible limitations have been discussed. To conclude, the present state of research suggests challenges of three kinds.

- **Natural design**
  Further research is needed to find natural designs for argumentation software. Not only must natural designs be explored and invented but also there is an urgent need to prove the level of naturality of these designs. An important way of proving naturality would be in terms of further user evaluation studies.

  One important issue is the style of interaction with a diagramming tool. General experience and user studies of the ArguMed system (Verheij, 2005b) suggest that interaction based on argumentation moves is more natural than interaction based on the graphical elements themselves.

  A further issue concerning natural design has to do with the way in which the boxes-and-arrows are arranged on the screen. For instance, the ArguMed software uses an arrangement algorithm in

which the argumentative statements are vertically ordered; no statements appear side by side. In this respect, the arrangement in ArguMed resembles an argumentative text that is a sequence of sentences. However, it might well be that other arrangements in which the horizontal arrangement of statements is used, is more natural. It may even be that assigning meaning to the arrangement of statements (as in Toulmin's diagram) can increase naturalness. As yet, comparative research answering such questions has not been systematically performed.

The fact that lawyers' arguments often take the form of oral or written texts leads to a third issue, namely how diagramming approaches can be connected with text-based approaches. The ArguGuide system provides preliminary results in this connection.

- **Usefulness**

  There is a need for further investigation into niches of usefulness for argumentation software, and of proof of the usefulness for these niches. Examples of such possible niches are argument analysis, argument drafting, skills training, argument presentation, case management, decision making and evidence investigation.[6] Notwithstanding their increasing appeal, where argumentation tools turn out to provide a substantial amount of added value is as yet largely unknown.

  Increased usefulness may require an extension of the expressiveness allowed by boxes-and-arrows diagramming, for instance the representation of variables and their instances, properties of rules and the weighing of reasons. Extended expressiveness should in principle extend usefulness. However, there is a risk of decreased usefulness in practice. Furthermore, there is interaction with the previous challenge: increased expressiveness might decrease naturalness.

- **Content**

  A final challenge is to include content-oriented support in argumentation software. Possible approaches to the providing of content in argumentation software have hardly been touched upon. The inclusion of argumentation schemes is a relevant development in this respect. Also the approach underlying the ArguGuide tool discussed in Section 3.4 seem promising in this respect.

For each of these challenges, formal empirical studies would be excellent ways of proving to what extent they have been met. But perhaps commercial success would be an even better way of proving the value of argumentation software.

Funding

REFERENCES

BEX, F. J., PRAKKEN, H. & VERHEIJ, B. (2006). Anchored narratives in reasoning about evidence. *Legal Knowledge and Information Systems. JURIX 2006: The Nineteenth Annual Conference* (ed. T. van Engers), pp. 11–20. Amsterdam: IOS Press.

BEX, F. J., PRAKKEN, H. & VERHEIJ, B. (2007). Formalising argumentative story-based analysis of evidence. *The 11th International Conference on Artificial Intelligence and Law. Proceedings of the Conference*, pp. 1–10. New York: ACM Press.

---

[6] Bex *et al.* (2006, 2007) discuss the connection between stories and argumentation in evidential reasoning. They provide a diagramming method of the boxes-and-arrows style.

CHAMBERS, E. (1728). *Cyclopaedia, or, an Universal Dictionary of Arts and Sciences*. London. http://digital.library.wisc.edu/1711.dl/HistSciTech.Cyclopaedia. Accessed on 8 February 2007.

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

DWORKIN, R. (1978). *Taking Rights Seriously. New Impression with a Reply to Critics*. London: Duckworth.

HAGE, J. C. (1996). A theory of legal reasoning and a logic to match. *Artificial Intelligence and Law* 4, pp. 199–273.

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

HITCHCOCK, D. L. (2006). Informal logic and the concept of argument. *Philosophy of Logic* (eds. D. Jacquette, D. M. Gabbay, P. Thagard & J. Woods). Amsterdam: Elsevier Science.

HITCHCOCK, D. L. & VERHEIJ, B. (eds.) (2006). *Arguing on the Toulmin Model. New Essays in Argument Analysis and Evaluation* (Argumentation Library: Vol. 10). Dordrecht: Springer-Verlag.

OSKAMP, A. & LAURITSEN, M. (2002). AI in law practice? So far, not much. *Artificial Intelligence and Law* 10, pp. 227–236.

POLLOCK, J. L. (1987). Defeasible reasoning. *Cognitive Science* 11, pp. 481–518.

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

REED, C. & ROWE, G. (2005). Translating Toulmin diagrams: theory neutrality in argument representation. *Argumentation* 19 (3), pp. 267–286.

REITER, R. (1980). A logic for default reasoning. *Artificial Intelligence* 13, pp. 81–132.

REITER, R. (1987). A logic for default reasoning. *Readings in Nonmonotonic Reasoning* (ed. M. L. Ginsberg), pp. 68–93. Los Altos (California): Morgan Kaufmann Publishers.

TOULMIN, S. E. (1958). *The Uses of Argument*. Cambridge: Cambridge University Press.

VAN DEN BRAAK, S. W., VAN OOSTENDORP, H., PRAKKEN, H. & VREESWIJK, G. (2006). A critical review of argument visualization tools: do users become better reasoners? *Workshop Notes of the ECAI-06 Workshop on Computational Models of Natural Argument (CMNA-06)* (eds. F. Grasso, R. Kibble & C. Reed), pp. 67–75. http://www.cs.un.nl/people/susanb/publications/ECAI06_Long_Paper_BraakEtAl.pdf. Accessed on 8 February 2007.

VERHEIJ, B. (1996). *Rules, Reasons, Arguments. Formal Studies of Argumentation and Defeat*.

VERHEIJ, B. (2003a). Artificial argument assistants for defeasible argumentation. *Artificial Intelligence* 150 (1–2), pp. 291–324.

VERHEIJ, B. (2003b). DefLog: on the logical interpretation of prima facie justified assumptions. *Journal of Logic and Computation* 13 (3), pp. 319–346.

VERHEIJ, B. (2003c). Dialectical argumentation with argumentation schemes: an approach to legal logic. *Artificial Intelligence and Law* 11 (1–2), pp. 167–195.

VERHEIJ, B. (2005a). Evaluating arguments based on Toulmin's scheme. *Argumentation* 19 (3), pp. 347–371.

VERHEIJ, B. (2005b). *Virtual Arguments. On the Design of Argument Assistants for Lawyers and Other Arguers*. The Hague: TMC Asser Press.

VERHEIJ, B., HAGE, J. C. & VAN DEN HERIK, H. J. (1998). An integrated view on rules and principles. *Artificial Intelligence and Law* 6 (1), pp. 3–26.

VERHEIJ, B., HAGE, J. C., VAN DER MEER, T. & SPAN, G. (2004). *Vaardig met recht. Over casus oplossen en andere juridische vaardigheden (Skilful in the Law. On Case Solving and Other Legal Skills)*. The Hague: Boom Juridische Uitgevers.

WALKER, V. R. (2007). A default-logic paradigm for legal fact-finding. *Jurimetrics* 47 (2), pp. 193–244.

WALTON, D. N. (1996). *Argument Schemes for Presumptive Reasoning*. Mahwah (New Jersey): Lawrence Erlbaum Associates.