Toward Legal Argument Instruction with Graph Grammars and Collaborative Filtering Techniques

Niels Pinkwart[1], Vincent Aleven[1], Kevin Ashley[2], and Collin Lynch[3]

[1] Carnegie Mellon University, HCI Institute, 5000 Forbes Avenue,
Pittsburgh PA 15213, USA
{nielsp, aleven}@cs.cmu.edu
[2] University of Pittsburgh, School of Law, 3900 Forbes Avenue,
Pittsburgh PA 15260, USA
ashley@pitt.edu
[3] University of Pittsburgh, Intelligent Systems Program,
Pittsburgh, PA 15260, USA
collinl@cs.pitt.edu

Abstract. This paper presents an approach for intelligent tutoring in the field of legal argumentation. In this approach, students study transcripts of US Supreme Court oral argument and create a graphical representation of argument flow as tests offered by attorneys being challenged by hypotheticals posed by Justices. The proposed system, which is based on the collaborative modeling framework Cool Modes, is capable of detecting three types of weaknesses in arguments; when it does, it presents the student with a self explanation prompt. This kind of feedback seems more appropriate than the "strong connective feedback" typically offered by model-tracing or constraint-based tutors. Structural and context weaknesses in arguments are handled by graph grammars, and the critical problem of detecting and dealing with content weaknesses in student contributions is addressed through a collaborative filtering approach, thereby avoiding the critical problem of natural language processing in legal argumentation. An early version of the system was pilot tested with two students.

1 Introduction

The field of law is an established and interesting application area for AI. (e.g. Aleven, 2003; Ashley 1990; Bench-Capon et al., 1998; Walton 2002). Argument is central to the practice of law, and therefore training in the skills of argument and advocacy are essential parts of legal education. Although there is a variety of law-related intelligent tutoring systems (e.g. Munjewerff and Breuker 2001), there are still only few intelligent tutoring systems specifically designed for assisting students in the construction of legal arguments. Exceptions include CATO (Aleven 2003) and ArguMed (Verheij 2003). CATO takes an example-based approach to teach students to make arguments based on past cases; ArguMed focuses more on structural aspects and provides assistants that support users in creating visual representations for defeasible arguments.

To some extent, the small number of tutoring systems for legal argumentation can be explained by that fact that the underlying domain is ill-structured. Legal argumentation is a kind of natural language discourse that focuses on interpreting the meaning

M. Ikeda, K. Ashley, and T.-W. Chan (Eds.): ITS 2006, LNCS 4053, pp. 227 – 236, 2006.

of general legal concepts in light of specific facts. In contrast to well-structured domains like mathematics, for most tasks in legal argumentation there is no unambiguously defined "correct" solution which could be used as a basis for an ITS.

The ITS approach described in this paper aims at supporting students in studying examples of legal argumentation drawn from US Supreme Court transcripts of oral arguments. The goal is to help students understand the dialectic in which advocates propose and modify tests (i.e. decision rules) for a case and the Justices pose hypothetical fact situations to assess the merits of the proposed tests. The texts involved in this task are rather unstructured and involve a wide range of (legal and world) knowledge. Thus, they are not well accessible for an ITS without applying natural language processing (NLP) techniques, which would be very error-prone in the interpretive field of legal argumentation.

In the next sections, we first discuss the importance of tests and hypotheticals for understanding legal arguments, and propose a structured representation format together with a corresponding visualization. We then describe a novel approach for intelligent tutoring based on these argument structures, in which the system is capable of detecting several types of weaknesses (not restricted to purely structural ones) in the student's conceptions of legal arguments, while retaining a partially textual representation format but without involving NLP. The paper concludes with a description of studies planned with the ITS.

2 Diagrams to Visualize Court Argument as Hypothesis Testing

In US Supreme Court oral arguments, contending attorneys each formulate a hypothesis about how the problem should be decided with respect to a set of issues. They may propose a test and identify key points of the facts at hand on which the issue should turn. The Justices test those hypotheses by posing hypothetical scenarios. These scenarios are designed to challenge the hypotheses' consistency with past decisions and with the purposes and principles underlying the relevant legal rules. These oral arguments provide interesting material for legal educators. They are concentrated examples of many conceptual and reasoning tasks that occur in Socratic law school classrooms. As discussed in Rissland (1989) and Ashley (1990), the oral arguments illustrate important processes of concept formation and testing in the legal domain. As such, studying the transcripts of these arguments and the contained process of test and hypothetical proposition and modification is a valuable task for beginning law students. Yet, this task is quite difficult for them due to the complexity of the argument.

One idea to overcome these problems is to augment the textual documents with structured graphical representations that express the argument structure explicitly, thereby providing data usable by an underlying intelligent support system. In general, the use of graphical representations to support argumentation is not a new approach. Suthers and Hundhausen (2003) have shown that using graph structures can support group argumentation processes (e.g., argument graphs invite relating parts to each other), and Van Gelder (2002) shows that reasoning with graphical representations can indeed be effective in strengthening critical thinking skills (measured by pre/post gains compared to traditional teaching methods). In the legal domain, Carr (2003) has used Toulmin schemas (Toulmin 1958) for collaborative legal argumentation, and the

Araucaria system (Reed and Rowe 2004) employs visual premise/conclusion structures. ArguMed (Verheij 2003) provides intelligent feedback through an argumentation "assistant" that analyzes structural relations between contributions in diagrams. Out of the three, only Carr (2003) conducted an empirical evaluation, but does not report whether his system caused learning gains. In summary, though a lot of promising general approaches for graphically supporting argumentation exist, current literature does not show much evidence for the educational effectiveness in the domain of legal argumentation.

In contrast to the approaches and systems referred to before, we recommend a novel special-purpose argument representation geared toward a particular kind of argumentation process in which a normative rule (or "test") is proposed, tested, and "debugged," primarily by means of hypotheticals. The main ontological categories in our argument representation are, simply, tests and hypotheticals. The representation can be used to track how attorneys modify their proposed tests to handle the hypothetical cases presented by the justices. It does not have the wide applicability of a general representation (such as Toulmin schemas), but its more specific ontological categories may help students interpreting argumentation processes, e.g. by focusing their attention on the relevant information. Similar to the approach adapted in Araucaria (Reed and Rowe 2004), we allow the student to explicitly relate argument structures to the textual transcript of the oral argument using simple markup techniques. There is evidence that students indeed make use of such markup functions if the system makes it easy to do so (Farzan and Brusilovsky 2005).

3 Three Mechanisms for Intelligent Support

Figure 1 shows a screenshot of the prototype system we implemented using the Cool Modes framework (Pinkwart 2005). The left side of the figure contains the transcript of the oral argument in a case called Lynch vs. Donnelly, 465 U.S. 668 (1983). At the bottom left, there is a palette with the elements (tests, hypotheticals, current fact situation) and relations (test modification, distinction of hypothetical, hypothetical leading to test change, general relation) that the student can use to construct via drag and drop mechanisms a graphical representation of the argumentation in the transcript. The workspace on the right side of the figure contains the argument representation. Figure 1 shows the result of a third year student's system usage within an exploratory study. The diagram records a variety of hypothetical cases presented by the Justices (five in total) and also contains the attorney's responses to these hypotheticals, in which he distinguished them from the facts at hand or by formulating tests that should cover the hypothetical and the problem.

As argued, rules which are guaranteed to detect errors in the student's argument graphs are virtually impossible, as there are no "ideal solutions" in the ill-structured domain of legal argumentation. However, the student's conception of the argument may have *weaknesses* (in the sense of *potential* problems) that can be classified into several types. Using the authentic student solution of the exploratory study as an example, the next subsections describe these different types of weaknesses, their detection within argument graphs, and how the intelligent tutoring system we have implemented responds to these detected "weak spots".

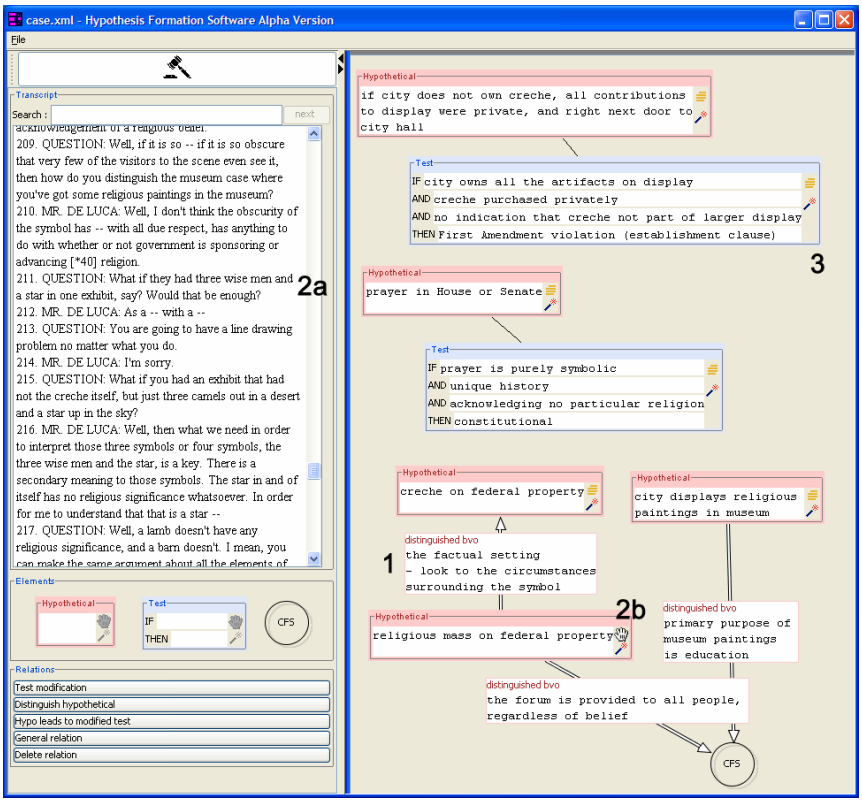

Fig. 1. Example of a graphical argument model

3.1 Structural Weaknesses

First, the argument representation created by a student can have *structural* weaknesses. Examples of weaknesses of this type are isolated elements in the argument graph, empty text fields, or the absence of any test element in the workspace. Figure 1 illustrates two more advanced structural weaknesses: only two out of the five hypotheticals are explicitly related to a test, the other three are not. Since attorney's often respond to hypotheticals by posing a new test, or a modification of an existing test, it may be that the student has overlooked (or misunderstood) a formulation of a test that appears in the transcript. In addition, the location marked (1) in the figure shows that the student distinguished two hypotheticals from each other. Attorneys and judges might well do this in an argument, but typically a hypothetical should also be related to the current facts in some way, e.g. through distinction. Since the student did not add links to the diagram to represent these relations, this part of the graph is a good candidate for system feedback to the student. Section 3.4 discusses how our ITS comments on these weaknesses, taking into account the remaining uncertainty that is based on weaknesses being just indications of *possible* errors.

Since structural weaknesses are related to the abstract structure of the argument only, they can be detected by logical formalisms that ignore both the content of the diagram text boxes and the markups of the transcript. Approaches for intelligent tutoring based on graphical argument structures are not new – early work in this field has been done by Paolucci, Suthers and Weiner (1996), who made use of syntax rules and an expert knowledge base to check argument graphs. Our proposed approach avoids "expert solutions" and model tracing, and relies on a graph grammar (Rekers and Schürr 1998) to analyze argument structures. The grammar consists of a set of terminal symbols T, a set of nonterminals N, a start symbol S, and a set of production rules. Both terminal and nonterminal symbols can have attributes, and a production rule is a tuple (L,R) of graphs over T∪N, which can be applied to a graph G that contains a subgraph M_L which matches L. The result of the rule application is a graph G' = G ∪ R \ M_L. Thus, a rule application replaces the subgraph that matches the left side of the rule with the graph in the right side.

We use the grammar to check the diagrams created by students for properties that represent structural argument weaknesses. Compared to other formalisms for "attribute value checking" which underlie many constraint-based tutors, the grammar based approach we propose is much better adapted to the graph structures we employ. A further advantage of the formalism is its declarative character: rules can easily be specified (cf. examples below) and applied in the system as parameters of the generic parsing algorithm. This avoids the need of programming a complex graph algorithm for each single property of the diagram that one wants to check.

The grammar contains two types of rules: first, "construction oriented" rules model the process of building argument graphs. The following rule illustrates this and covers the situation that a "test" element is added to an empty workspace (in this case, the test gets assigned the value "unchanged" for the "version" attribute):

```
L = < {S},∅ >
R = < {TEST},∅ >
TEST.version = "unchanged"
```

In addition, "feedback oriented" rules directly express a specific weakness and thus enable the system to produce well-defined feedback. The following is an example of such a "feedback oriented" production rule, which can detect the structural weakness of "hypothetical distinction without relation to the facts" that was discussed in relation to Figure 1:

```
L = < {HYPO1,HYPO2,W},{Distinguish_from(HYPO1,HYPO2)} >
R = < {HYPO1,HYPO2},{Distinguish_from(HYPO1,HYPO2)} >
HYPO2.connection = "false"
W.type = "isolated_hypo_distinguished_from_hypo"
W.locations = {HYPO1,HYPO2}
```

The right side of the rule matches the student solution of Figure 1 by identifying HYPO1 with the "religious mass on federal property" hypothetical and HYPO2 with the "crèche on federal property" element (note that the "connection" attribute of HYPO2 is used to express its lack of relation to the facts). The nonterminal node W in the left side of the rule represents the detected weakness.

3.2 Context Weaknesses

The second weakness type can be characterized as *context* weaknesses. These essentially deal with the relation between the argument graph and the transcript. Even if an argument diagram has no structural weaknesses, the relation between the elements in the diagram to the source material (i.e., the transcript) as expressed through the markups can reveal problems. Examples of context weaknesses are a lack of evidence for the tests/hypotheticals in the diagram (missing nodes or links), important passages of the transcript that are referenced in the diagram but with (apparently) the wrong element type (e.g., a test being marked up as a hypothetical), or seemingly irrelevant parts of the transcript being marked up. Figure 1 illustrates two of these weaknesses. In (2a), the transcript lines "What if they had three wise men and a star in one exhibit, say? Would that be enough?" contain a hypothetical posed by a judge, but the student's solution does not refer to it in any way. Also, the hypothetical (2b) is not linked to the transcript (visible through the hand symbol in the right corner of the element).

The same graph grammar formalism that is used to detect structural weaknesses (described above) is also applicable for detecting context weaknesses, which obviously is an advantage on the technical level. We make use of node and edge attributes in the grammar rules to represent constraints on the links that are created between the transcript and the elements in the graph. A context rule for the most important context weakness (missing link to important part of transcript), can be specified as follows:

```
L = < {S},∅ >
R = < {HYPO},∅ >
HYPO.link = 211
```

This rule is comparable to the start rule in the "structural weaknesses" part, differing only in that it requires specific elements to be present in the argument graph. This rule requires a student to mark up line 211 of the transcript and link it to a hypothetical. Similar rules can be formulated to explicitly declare "irrelevant" parts of the transcript that should not be marked up.

Taken together, structural and context rules allow a teacher to specify in detail a particular test/hypothetical structure linked to well-defined parts of the transcript. However, we are not advocating an approach in which the student's graph is compared against an expert solution. Due to the ill-structuredness of the legal domain, it is not possible to define a small set of "correct" solutions. Instead, we use the graph grammar formalism to partially specify solutions (e.g., only the two most important test versions and six central hypotheticals are required to be marked up by students).

3.3 Content Weaknesses

Finally, the *content* of the textual elements created by the student can be inappropriate, even if the overall argument structure is good and related to the transcript in a reasonable way. Students may well have difficulties in understanding, e.g., the essence of a proposed test, as evidenced by a poor paraphrase in the corresponding test node they add to the graph. Obviously, this type of weakness is hardest to check, since it involves interpretation of legal argument in textual form. In addition, due to the ill-structured domain, student answers will not be simply either right or wrong, but

instead have a certain quality in terms of a number of criteria. For instance, location (3) in Figure 1 contains the student's description of the test proposed in the argument. For a general reader (and also for an ITS), it is hard to tell if this is an adequate summary of the test or not. The graph structure and peers working on the same task can help.

Our two-step approach to address the problem is NLP-free and involves a technique known as "collaborative filtering" (Konstan and Riedl 2002) – we make the assumption that a larger group of students works on the same task, either individually or in small groups. In our variant of the collaborative filtering method, students are asked to rate samples of other's work relative to their own work. The system then combines the ratings into an overall score and thus "filters" for quality. More specifically, for selected parts of the transcript (i.e., parts where a test is mentioned), after a student has created a corresponding element in the graph, he is first presented with a small number of alternative answers (given by peers) and asked to select all those he considers *similar* to his own answer. Then, the student gets a second selection of answers (some known to be good, some known to be of poor quality, some given by peers) and is asked to select all those he considers *at least as good as* his own. The system then uses a combination of similarity and recommendation ratings to compute a heuristics of the quality of student answers.

A base rating b_x of an answer given by student x can be calculated based on the recommendations given by x. If the student had n answers to choose from, and the ones he selected as being at least as good as his own had a quality measure $q_1, \ldots, q_k$ (0 for very bad, 1 for very good, see below for the calculation of quality measures for peer answers), while those he did not recommend had quality measures $q_{k+1}, \ldots, q_n$, then b_x can be calculated as

$$b_x = \frac{1}{n}\left(\sum_{i=1}^{k} q_i + \sum_{i=k+1}^{n} (1 - q_i)\right)$$

Figure 2 shows a window presented to the student for the base rating calculation. In the figure, three answers can be selected as "at least as good as his own" by the student. If the three available answers have quality ratings of 1, 0.8, and 0.3 (i.e., two good ones and two bad ones), and the first two have been selected like the figure shows, then the base rating for the student's answer is $b_x = 0.33*(1+0.8+0.7) = 0.825$. Considering the good quality of the test description provided by the student, this base rating is acceptable as an initial value.

The base rating b_x measures in how far a student can recognize good answers and thus serves as a heuristic of his own answer's quality, but does not rate the answer the student has actually typed in himself. In our approach, the base rating is therefore supplemented by two other ratings which measure the quality of what the student has actually typed in. First, the *similarity estimations* given by the students are used. If students with good own ratings rate another solution as similar, this raises the rating of the peer solution (the peer rating will also be reduced based on similarity with poor solutions). Second, the *recommendations* that a student's answer receives by his peers are used, with recommendations by good students having a higher impact. The overall quality q_x of a student answer is then calculated as the weighted average of b_x and the other two measures. The weights of the peer-dependent ratings are based on the

number of selection options that peers had. This takes into account the importance of peer's opinions while at the same time eliminating the cold start problem (how to handle the first users of the system, before peer ratings are available?) through the inclusion of relations to known correct/incorrect solutions, which feed into b_x.

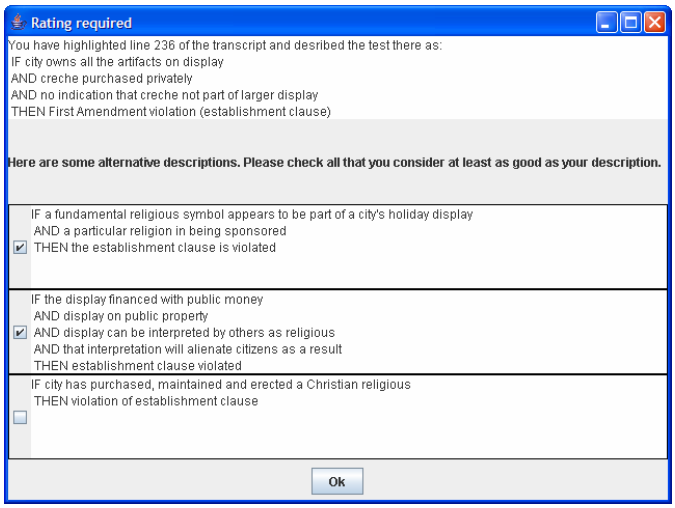

Fig. 2. Similarity rating dialog

3.4 Tutor Feedback

The previous sections described how different kinds of weaknesses can be detected in the student's argument graphs. Having detected a weakness, the question is how the system should react to it. Our notion of weakness includes the possibility of "false alarms": a student's solution can be of high quality and still cause a tutor intervention. This seems inevitable in an ill-structured domain, where correctness is hard to define even for human domain experts, for example: Does the fact that the student distinguishes two hypotheticals (see location 1 in Figure 1) without relating the hypotheticals to the current fact situation indicate that the student did not understand the role of hypotheticals in the argument, or was this just a wrong use of graphical elements?

Since these questions cannot be answered in a general way by an ITS, it does not make sense for an ITS to use most of the detected weaknesses as a basis for telling users directly that they were wrong in their answer. However, following the idea of weaknesses as the presumably weak parts of student's work, it makes sense to use them as tailored and personalized self-explanation prompts by inviting the student to re-think and explain these parts of his work. Self-explanation has been shown to be effective in many domains, including ill-structured ones (Schworm and Renkl 2002). Table 1 shows some of the weaknesses that were identified within this article together with related short versions of self-explanation prompts.

Table 1. Examples of self-explanation prompts

Weakness Description	Type	Self-explanation prompt (short version)
Isolated hypothetical distinguished from hypothetical	structural	In your solution, the hypotheticals H1 and H2 are distinguished from each other. Yet, hypothetical H2 is not related to any test or the current fact situation. Please explain why you did so, either in free text or by modifying the diagram.
Important part of transcript not marked	context	Please look at this part of the transcript (scroll to line L) and explain its role within the argument.
Low quality rating of contribution	content	Please reexamine what you marked here (scroll to line L) and explain it again.

4 Conclusions and Outlook

The approach as presented in this paper is designed to support first-year law students in learning legal argumentation skills by having them create graphical models of argument transcripts, and presenting them feedback on the weaknesses in their models. The ITS used to generate this feedback is based on two formalisms, which enable a heuristic check of student answers for different types of weaknesses: a graph grammar formalism and a collaborative filtering technique. It does not make use of NLP, which can be considered an advantage in the highly interpretive and ill-structured domain of legal argumentation, but nevertheless is able to give content-related feedback. Furthermore, the approach requires only very minimal system-side knowledge about specific legal cases, which facilitates using the system with a new transcript.

The pilot studies we conducted essentially confirmed the suitability of the ontological categories and the graphical representation format. Based on these, further research will try to find empirical evidence for the effectiveness of the presented tutoring approach, both compared to control groups that make use of the diagram tool without feedback, and also to groups that work traditionally. In particular, we are interested in "fine tuning" the selection of feedback prompts and the collaborative filtering mechanism in terms of which peer answers are best to present to a student.

Acknowledgements

This research is sponsored by NSF Award IIS-0412830. The contents of the paper are solely the responsibility of the authors and do not necessarily represent the official views of the NSF.

References

1. Aleven, V. 2003. Using Background Knowledge in Case-Based Legal Reasoning: A Computational Model and an Intelligent Learning Environment. *Artificial Intelligence* 150: 183-238.
2. Ashley, K. 1990. *Modeling Legal Argument: Reasoning with Cases and Hypotheticals*. Cambridge MA, MIT Press/Bradford Books.

3. Bench-Capon, T., Leng, P., and Staniford, G. 1998. A computer supported environment for the teaching of legal argument. *J. of Information, Law & Technology* 3
4. Carr, C. 2003. Using Computer Supported Argument Visualization to Teach Legal Argumentation. In *Visualizing Argumentation*, 75-96. London, Springer.
5. Farzan, R., and Brusilovski, P. 2005. Social Navigation Support through Annotation-Based Group Modeling. In Proc. of UM, 387-391. Berlin, Springer.
6. van Gelder, T. 2002. Argument Mapping with Reason!Able. *The American Philosophical Association Newsletter on Philosophy and Computers* 85-90.
7. Konstan, J., and Riedl, J. 2002. Collaborative Filtering: Supporting social navigation in large, crowded infospaces. In *Designing Information Spaces: The Social Navigation Approach*, 43-81. Berlin: Springer.
8. Muntjewerff, J., and Breuker, J. 2001. Evaluating PROSA, a system to train solving legal cases. In Proc. of AIED, 278–285. Amsterdam, IOS Press.
9. Paolocci, M., Suthers, D., and Weiner, A. 1996. Automated Advice-Giving Strategies for Scientific Inquiry. In Proc. of ITS, 372 - 381. Berlin, Springer.
10. Pinkwart, N. 2005. *Collaborative Modeling in Graph Based Environments*. Berlin, dissertation.de Verlag.
11. Reed, C., and Rowe, G. 2004 Araucaria: Software for Argument Analysis, Diagramming and Representation. *International Journal of AI Tools* 14:961-980.
12. Rekers, J., and Schürr, A. 1997. Defining and parsing visual languages with layered graph grammars. *Journal of Visual Languages and Computing* 8:27-55.
13. Rissland, E. 1989. Dimension-Based Analysis of Hypotheticals from Supreme Court Oral Argument. In Proc. of AI & Law, 111-120. New York, ACM Press.
14. Schworm, S., and Renkl, A. 2002. Learning by solved example problems: Instructional explanations reduce selfexplanation activity. In Proceedings of the 24th Annual Conference of the Cognitive Science Society, 816-821. Lawrence Erlbaum.
15. Suthers, D., and Hundhausen, C. 2003. An Experimental Study of the Effects of Representational Guidance on Collaborative Learning Processes. *Journal of the Learning Sciences* 12:183-218.
16. Toulmin, S. 1958. *The Uses of Argument*. Cambridge University Press.
17. Verheij, B. 2003. Artificial argument assistants for defeasible argumentation. *Artificial Intelligence* 150:291-324.
18. Walton, D. 2002. *Legal Argumentation and Evidence*. Penn State Press.