Generating Dialectical Examples Automatically*

Kevin D. Ashley and Vincent Aleven
University of Pittsburgh
Intelligent Systems Program,
School of Law, and Learning Research and Development Center
Pittsburgh, Pennsylvania 15260

Abstract

We identify and illustrate five important kinds of Dialectical Examples, standard configurations of cases which enable an arguer to justify rhetorical assertions effectively by example. Our computer program generates Argument Contexts, collections of cases that instantiate Dialectical Examples from an on-line database of cases according to a user's general specifications. The Argument Context generation program provides a human or automated tutor a stock of Dialectical Examples to teach novice advocates (first year law students) how to recognize, carry out and respond to the associated rhetorical moves. Although generating such examples is very hard for humans even when dealing with small numbers of cases, our program generates and organizes such examples quickly and effectively. In a preliminary experiment, we employed program-generated Argument Contexts manually to teach basic argument skills to first year law students with good results. Our ability to define such complex examples declaratively in terms of logical expressions of Loom concepts and relations affords a number of advantages over previous work.
Topic: argument from experience, law, education

Introduction

When an advocate justifies an assertion by referring to past cases or examples, she employs a variety of standard techniques involving recognized configurations of cases to make her rhetorical points effectively. We call such configurations "Dialectical Examples" and identify five important types that advocates employ as building blocks of more complex arguments. While Dialectical Examples are valuable assets for expert advocates and teachers of argumentation skills, finding the right configurations of cases is hard, even when one uses current computerized information retrieval services. There are tight constraints on how the cases must be related and many combinations of cases to consider. Adopting a methodological viewpoint inspired by Clancey [Clancey1983], we have reexamined previous work on HYPO, a case-based legal expert system [Ashley1990, Ashley1991], to discover how to reorganize the knowledge to support a tutoring system that can teach novices to analyze problems and construct legal arguments. The tutoring task requires making explicit certain information that was previously implicit: an expert's knowledge of how to construct, employ, and respond to Dialectical Examples in support of argument positions.

We have built a program that efficiently generates "Argument Contexts", graph-like configurations of cases that instantiate the five types of Dialectical Examples, according to the user's specifications. The program queries an argumentation knowledge base, implemented in Loom [MacGregor1988], comprising definitions for argumentation concepts and representations of aspects of 26 legal cases. We have conducted a preliminary experiment to evaluate whether program-generated Argument Contexts can be used to advantage in teaching basic argument moves to first year law students. This work advances AI research on example generation and argumentation [Rissland and Soloway1980, Rissland *et al.*1984, Suthers and Rissland1988, McGuire *et al.*1981, McCarty and Sridharan1981]. Argument Context generation facilitates teaching classification concepts (such as legal claims, defined below) which are not definable neatly in terms of necessary and sufficient conditions, unlike those dealt with by Collins and Winston, [Collins and Stevens1982, Winston1975], and where the instances do not support the construction of isomorphic analogical mappings (See, e.g., [Gentner1983]).

Dialectical Examples

Arguing by analogy to past cases or examples can be modeled in terms of cases, outcomes and factors [Ashley1990, Ashley1991]. In a legal domain, cases are disputes between the plaintiff, the side that initiates a lawsuit, and the defendant. We will refer to a current dispute as the "current fact situation" or "cfs". Past cases are disputes in which a court has previously decided that a plaintiff won or lost. The plaintiff in the

*This work is supported by a NSF Presidential Young Investigator Award and a grant from the National Center for Automated Information Retrieval.

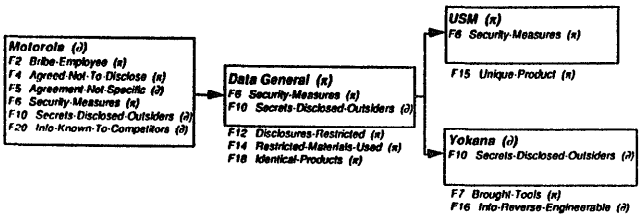

Figure 1: Four-case "conflict resolution" Argument Context

current fact situation asserts some legal claim against the defendant, such as breach of contract or trade secret misappropriation, the domain of this work, and makes arguments to convince a court that its claim is valid. We assume that in any dispute involving a legal claim, one may identify factors, collections of facts that strengthen the plaintiff's or the defendant's side. Typically, some of a dispute's factors favor the plaintiff and some the defendant. Although an advocate may argue that the factors in its favor outweigh any factors favoring the opponent, he needs to cite some legal authority in support of this conclusion. Unfortunately, however, the law has no authoritative scheme for assigning weights to factors. In general, it is not appropriate for a legal advocate to employ statistical arguments [Ashley and Rissland1988]. Instead, there are a variety of rhetorical techniques, employing Dialectical Examples, for justifying assertions about factors and cases.

We have identified five general kinds of Dialectical Examples that have utility in making and testing arguments, or in teaching these skills. Each involves a configuration of cases that enables an arguer to make a rhetorical point effectively by drawing conclusions from a symbolic comparison of cases. Each is illustrated below with an Argument Context generated on demand by our program from a database of cases. The different types of Dialectical Examples include:

1. Representative examples
2. Conflict Resolution examples
3. Refutational comparisons (i.e., counterexamples)
4. *Ceteris paribus* comparisons
5. Coherence examples

Before we introduce the various Dialectical Examples, we illustrate some of the basic elements of our model of case-based argument. Figure 1 shows an Argument Context in the form of a Claim Lattice [Ashley1990, Chapters 5,8]. It is a graph, each node of which represents at least one case. The root node represents a current fact situation, the *Motorola* case, to which six factors apply, three of which favor plaintiff (π) and three the defendant (δ). These factors all deal with a claim for trade secret misappropriation. In the *Motorola* case, the problem situation (or cfs), the plaintiff took certain security measures to protect its confidential information (F6), including securing nondisclosure agreements from its former employees (F4) involved in this suit. The employees left Motorola to work for the defendant corporation, allegedly bringing Motorola's secrets with them. These employees obtained substantial inducements like raises in making the move (F2), suggesting the possibility of a payoff for bringing plaintiff's secrets with them. On the other hand, favoring the defendant, plaintiff had allowed some of its secrets to be disclosed to outsiders (F10), the allegedly secret information was known to competitors (F20) and the employee nondisclosure agreements did not make clear exactly what information the plaintiff regarded as secret (F5). In treating *Motorola* as a cfs, we assume that its outcome has yet to be decided (in fact the defendant won). Each of the other nodes represents a past legal case, or precedent, also represented as a set of factors, but one to which some authority, a judge, has assigned an outcome, that is, the plaintiff either won or lost its legal claim. All of the cases share at least one factor with *Motorola* and are ordered in terms of the inclusiveness of the sets of factors the case shares with the cfs. *Data General* is "more on point" relative to the cfs than *Yokana* because *Yokana's* set of factors shared with *Motorola* is a proper subset of that of *Data General*. Each case may also have factors it does not share with the cfs. These unshared factors are listed below the node.

1. Representative examples. One of the simplest justifications of a factor's importance is a representative example in which the factor clearly contributed to the outcome of the case. We have identified three kinds of representative cases: *vanilla*, *simple conflict* and *packed* cases. Vanilla and simple conflict cases are examples that clearly represent the effect of a factor. A vanilla case is one in which at most two factors apply, both of which favor the winner of the case; it emphasizes those particular factors. A single conflict case is one where the factor of interest is consistent with the case's outcome (i.e., pro-winner) and all the other factors favor the side that lost. Such a case makes a dramatic demonstration of a factor's importance in overcoming all of the competing factors. Packed cases represent the effect of a collection of consistent factors, in other words, they are situations in which one of the parties had a very strong position. Such cases have a larger number of factors, all of which (or all but one of which) are consistent with the case's outcome. In tutoring, representative examples are useful to introduce students to their respective factors, vanilla cases because they are so uncomplicated, simple conflicts cases because they so strongly emphasize the effect of a particular factor, and packed cases because they are an economical way to introduce lots of factors in a concentrated form.

Citing a parade of representative examples for each of a side's strengths is a basic, but important, kind of legal argument. An advocate for the plaintiff in

the *Motorola* case of Figure 1 could justify an assertion that the facts associated with factors F2, F4 and F6, all of which favor plaintiff, justify a victory for its client, by citing some representative examples. For instance, *USM* is representative of cases where the pro-plaintiff *Security-Measures* factor justifies a victory for the plaintiff. The plaintiff's attorney could make a simple, but reasonable argument in favor of its client as a representative example of the effect of factor F6: "Where the plaintiff took measures to protect the security of its alleged trade secrets, as plaintiff Motorola did, it should win a claim for trade secret misappropriation, just as the plaintiff won in the *USM* case".

Packed cases can also be cited as representative examples, but are more likely to be distinguishable. The plaintiff in *Motorola* can also cite *Data General*, a packed case, as a representative example of the effect of factor F6 (as we see below, citing *Data General* has certain other rhethorical advantages), but in so doing, the plaintiff opens itself to a response by the defendant. *Data General* is a strongly pro-plaintiff packed case; three factors favoring plaintiff apply to it that do not apply to *Motorola* and offer an alternative explanation of the result. In responding to plaintiff's argument citing *Data General* as a counterexample, the defendant in *Motorola* could distinguish *Data General* by pointing out the unshared pro-plaintiff factors F12, F14 and F18[1].

2. Conflict Resolution examples support assertions that a set of factors all favoring one side is more important than another set of factors all favoring the opponent by showing a case with both sets whose outcome is consistent with the former. In Figure 1, *Data General* has the virtue that it accounts not only for some of the plaintiff's strengths in *Motorola* but also for some of the opponent's strengths; it resolves the conflict among factors F6 and F10. Though there is no legally authoritative weighting scheme to which the plaintiff can appeal to justify an assertion that *Security-Measures* is a more important factor than *Secrets-Disclosed-[to]-Outsiders*, *Data General* is an authoritative example that the conflict should be resolved in favor of the plaintiff.

For tutoring students about conflict resolution, Figure 1 may not be an ideal example; conflict resolution examples are more effective the more on point they are relative to the problem and the less distinguishable they are. Although *Data General* is a more persuasive case than *Yokana* in so far as it accounts for a more inclusive set of factors in *Motorola*, it is less persuasive to the extent that it can be distinguished from the cfs. On the other hand, since a tutor needs to teach students

[1] Within a particular context, certain differences among cases are salient and others not. Our model enables one to determine in context, which differences are, in fact, distinctions. See [Ashley1989] for more information.

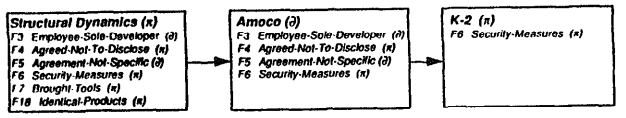

Figure 2: Three-case "trumping counterexample" Argument Context

to respond to a conflict resolver by distinguishing, Figure 1 may be a very good example. Our program enables a tutor to generate and select examples suitable for particular tutoring contexts. The user inputs the number of representative and conflict-resolving precedents that the Argument Context must contain. The program formulates an appropriate query and runs it. The user can then filter and rank the generated Argument Contexts according to criteria she deems useful, for instance that the conflict resolver has few distinctions.

3. Refutational or counterexamples are cases that refute an assertion. If an opponent asserts that a given set of factors necessitates victory for his side, one may refute that with a case where the factors applied but that side did not win. A given factor may have many refuting examples. Although statistical arguments are generally not acceptable, it would be appropriate to support an argument that a factor is not significant with a parade of refuting cases. The program searched for cases to refute the pro-defendant factors in *Motorola*. Only factor F5, the fact that the non-disclosure agreement was not specific, had more than one refutational example (it had two, not a dramatic argument.)

A trumping counterexample is a more contextually specific kind of refutational example, because it refutes an opponent's assertion that the set of shared factors associated with the cited precedent necessitates the same outcome [Ashley1990, Chapters 8, 9]. In essence, plaintiff trumps the defendant's case and refutes its point by citing a counterexample that is more on point (i.e., a trumping or more-on-point counterexample). For instance, in Figure 1, if the defendant in *Motorola* cited *Yokana* for the proposition that the plaintiff's disclosures to outsiders (F10) necessitate a victory for defendant, the plaintiff could refute the defendant's assertion by citing *Data General* as a counterexample. There, the plaintiff won despite the disclosures, where the plaintiff also took security measures (F6). As a trumping counterexample, *Data General* satisfies the additional constraint that it shares a more inclusive set of factors with the cfs than the cited case does.

Here, again, our program enables a tutor to generate and select instances of trumping counterexamples suitable for particular tutoring contexts. Like conflict resolvers, trumping counterexamples are more persuasive the more extra similarities they share with the

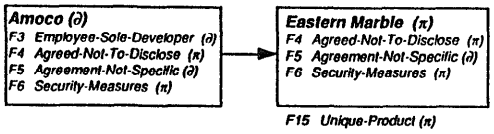

Figure 3: Two-case "*ceteris paribus*" Argument Context

cfs than the less on point case does and the less distinguishable they are from the cfs. For purposes of introducing students to the rhetorical uses of trumping, a better example is shown in Figure 2. Here, the program has retrieved 81 three-case trumping counterexamples and ranked them according to the criteria mentioned, enabling the tutor to select the example in the figure. The program is able to formulate queries for these Argument Contexts; the user simply specifies the number of cases.

From rhetorical and pedagogical viewpoints, the absence of refutational examples is interesting, too. If no case involving a particular factor had an outcome inconsistent with the factor, that makes an effective argument that the factor is very important. For instance, using the program, a student can discover that factor F20 in *Motorola* (the allegedly confidential information was known to competitors), has no refutational examples while there are three cases in the database of 26 cases where factor F20 applied and defendants won. Although the effectiveness of such an argument depends on the size of the sample and exhaustiveness of the search, the rhetorical concept can be taught with the help of the program.[2]

4. Ceteris paribus comparisons provide another way to support assertions that a factor is important by showing the difference the factor makes to the outcome of two cases that are equivalent but for that factor. Pursuant to this Dialectical Example, one shows pairs of cases where all things are equal except for the presence of a certain factor which accounts for the difference in outcomes. For instance, suppose an advocate wanted to substantiate an assertion that a defendant employee should win where that employee, while working for the plaintiff, was primarily responsible for inventing and developing the confidential information the plaintiff seeks to protect (factor F3 – *Employee-Sole-Developer*). Two cases, differing by the addition of just factor F3 where the defendant won the latter case, would be ideal. In figure 3, a comparison of *Amoco* and *Eastern Marble* substantiates the effect of factor F3, which, arguably, explains why the defendant won in *Amoco* even though that case is much like *Eastern Marble* where the plaintiff won.

[2]As illustrated above, the nature of the refutation and the assertions refuted may be of a variety of different types. An opponent's assertion that the *absence* of particular factors necessitates losing a claim may be refuted by a case where the side won but no such factors were present. Similarly, rules may be "broadened" by cases where the rule applies though certain supposed prerequisites are absent [Rissland and Skalak1989].

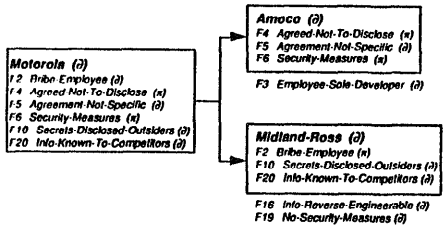

Figure 4: Three-case "*cover bases*" Argument Context

The program enables a tutor to select all of the pairs of cases in the database suitable for making *ceteris paribus* comparisons or to search for comparisons for a particular factor. The presence of additional differences may spoil a *ceteris paribus* comparison. Here, *Eastern Marble* also differs from *Amoco* in that it has factor F15, plaintiff's product was unique. Since perfect comparisons are relatively rare, our program's definition of a *ceteris paribus* comparison admits pairs with more than one difference. The comparisons are ranked according to effectiveness as measured in part by the absence of extraneous differences.

5. Coherence examples are case examples which stick together in some sense. Cohesiveness may be supplied by substantive or tactical considerations. Substantive coherence can be supplied by citing cases having a common theme or analytical rationale (we are planning to represent some of these aspects of cases.) For tactical reasons, an advocate may cite a set of conflict resolution cases that "cover the bases" in that they effectively counteract all of an opponent's strengths. That is, for all of the opponent's strengths, the advocate cites cases in which those strengths were overcome by as few of the advocate's strengths as possible. Such an argument shows that none of the opponent's strengths is fatal. For instance, the program generated the Argument Context of Figure 4 which covers the bases on behalf of the defendant in *Motorola*. *Midland-Ross* and *Amoco* were both won by the defendant, and together have all the pro-plaintiff factors that apply in *Motorola*. This rhetorical device supports a conclusion that despite the plaintiff's strengths in *Motorola*, despite the employee bribe (F2), the nondisclosure agreement (F4) and the security measures (F6), defendant should still win.

By presenting such program-generated examples, a tutor may teach students to seek a parsimonius cover, that is one in which each case takes minimal advantage of an advocate's strengths while the cases as a group effectively cover the opponent's strengths. At the same

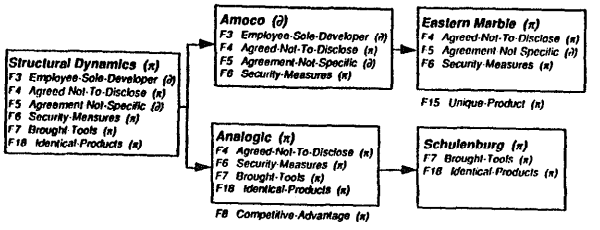

Figure 5: Five-case "select best case" Argument Context

time, a student needs to learn to cite cases that are more on point and less distinguishable. In Figure 4, for instance, both covering cases are distinguishable from the cfs.

Argument Contexts in Tutoring

These Dialectical Example types are building blocks of more complicated arguments. An advocate attempts to assemble some set of these examples, tailored to the specific problem and its factors. If the advocate believes that a particular set of factors are key to his case, for instance, he may cite some relatively non-distinguishable representative cases focusing on those factors, and, if possible, make some *ceteris paribus* comparisons to establish the factors' significance. If the advocate seeks to blunt the effect of the opponent's strengths, he may cover the bases with cases that show those strengths are not fatal and cite as on point a favorable conflict resolving case as possible. In planning arguments, an advocate will always be on the lookout for his opponent's trumping counterexamples.

Students need to learn rhetorical skills like recognizing and employing different kinds of Dialectical Examples and making reasonable choices among possible argument moves. We are designing a tutoring system to teach law students to learn such skills by presenting them with Argument Contexts, like those above, that present opportunities for employing the associated argument moves [Ashley and Aleven1991, Aleven and Ashley1992].

The tutoring method can be illustrated with Figure 5 which shows an Argument Context the program generated to teach an introductory lesson about: (1) recognizing and avoiding trumping counterexamples , (2) the minimum criteria for citing a case , and (3) the fact that the more on point a case, the better it is to cite. Given this Argument Context, a tutor can ask a student to select the best case to cite for the plaintiff. A student's answer reveals a lot about his or her understanding of arguing with cases. More specifically, although *Analogic*, *Eastern* and *Schulenburg* are all citable for plaintiff, *Analogic* is the best precedent. It is better than *Eastern*, because *Eastern* can be trumped by *Amoco*, whereas *Analogic* cannot be trumped (1). *Analogic* is better than *Schulenburg* because it is more on point, and according to (3), a more on point precedent is better. *Analogic* is better than *Amoco*, because *Amoco* cannot be cited on behalf of the plaintiff – it does not satisfy the minimum requirements (2). If the student misses the trump, the program can present instances of trumping counterexamples, such as the Argument Context of Figure 2, to follow up. To generate the Argument Context shown above, the user only needed to specify the Issues that she wanted the Argument Context to bring up; the program itself formulated and ran the query.

Generating such pedagogical examples, however, is a difficult task to perform by hand. A law school legal methods instructor agreed that the Argument Context of Figure 5 would be useful for pedagogical purposes but estimated that it would take hours or days to discover a set of such cases even using available on line retrieval services [Saunders1991]. Even a human tutor already familiar with the opinions of twenty or thirty cases would find it very difficult mentally to construct Argument Contexts. The criteria are abstract and there are many combinations of cases to consider, few of which are satisfactory examples.

Generating Argument Contexts

We have designed a case representation and program that generates the Argument Contexts shown in this paper, and others, automatically in seconds. The user inputs specifications for the Argument Contexts by entering parameters for certain standard Argument Contexts or a query like the one illustrated below. The program outputs instantiated Argument Contexts satisfying the query and enables the user to inspect the generated Argument Contexts, filter and rank them according to various useful criteria, and save them to a file.

The program generates Argument Contexts by querying a knowledge base for case-based argumentation that we have implemented using the knowledge representation system Loom [MacGregor1988]. Loom is a structured inheritance system, or KL-ONE-style system [Woods and Schmolze1990]. To represent knowledge in Loom, one provides definitions for concepts and relations, and, asserts facts about individuals, the instances of the concepts.

Loom provides a deductive query facility that allows one to retrieve most logical consequences of the facts and definitions in the knowledge base. Query expressions are, roughly speaking, first-order logic formulae. An example of a query is given below. Loom evaluates queries by a process of exhaustive search. The query language can also be used to state definitions of concepts and relations.

The knowledge base contains definitions, most of them in Loom's query language, of many important concepts and relations involved in case-based argumentation as well as some pedagogical concepts including the following:

Primitives	Pedagogical concepts	Argument concepts
Case	Vanilla Case	Shared Factor
Factor	Packed Case	Citable
Side (π or δ)	Conflicting Factors Case	More On Point
Outcome	Unordered	Best Case To Cite
Applicable Factor		Trumping Counterexample
Favors		Relevant Difference

The program's knowledge base contains facts that represent important aspects of the 20 factors, and 26 cases that we work with. For each factor, the side that it favors is recorded, for each case, its set of applicable factors and the side that won. Below, we display a query that retrieves two-case "cover the bases" Argument Contexts; the three-case Argument Context of Figure 4 was generated by a similar query. In English the query says: "Retrieve cases `?cfs` and `?c1`, such that `?c1` is won by side `?s` and such that all factors in `?cfs` that favor `?s`'s opponent are covered by `?c1` (that is, apply in `?c1`); there must be at least one such factor."

```
(RETRIEVE (?CFS ?C1 ?S)
   (:AND (CASE ?CFS)
       (TSM-PRECEDENT ?C1)
       (NEQ ?C1 ?CFS)
       (OUTCOME ?C1 ?S)
       (:FOR-SOME ?F
         (:AND (FACTOR ?F)
               (APPLICABLE-FACTOR ?CFS ?F)
               (FAVORS ?F (OPPOSITE ?S))))
       (:FOR-ALL ?F
          (:IMPLIES
             (:AND (FACTOR ?F)
                   (APPLICABLE-FACTOR ?CFS ?F)
                   (FAVORS ?F (OPPOSITE ?S)))
             (APPLICABLE-FACTOR ?C1 ?F)))))
```

Notice that the query refers to concepts and relations that are defined in our knowledge base. This query returned 11 Argument Contexts in 3.5 seconds. The four-case Argument Context of Figure 1, together with 8 similar ones, was generated by the program in 6 seconds. The query for the three-case Argument Contexts returned 4 Argument Contexts in 22.9 seconds. (Most queries run faster.) The program is described more fully in [Ashley and Aleven1991, Aleven and Ashley1992].

For pedagogical use, Argument Contexts need to satisfy special constraints that facilitate teaching students how to use them. They may present some especially clear cut examples of the above, or situations presenting some combination of the above and posing a choice of which argument move to make. One does not have to express such preferences as "hard" constraints in a query, but can use the program's filtering, ranking and sorting facilities to find the best pedagogical examples. In Section 2, we gave some brief illustrations of the filtering process.

Tutoring Experiment

We have conducted a preliminary experiment to assess empircally whether program-generated Argument Contexts make useful examples in tutoring argument skills.[3] An "experimental group" of three first-year law students and a two-student "control group" took a one hour, written pre-test to provide a baseline indication of their case-based argumentation skills. Kevin Ashley manually conducted a seventy-five minute tutorial session for the experimental group. Then all five students took a one hour, written post-test to assess any changes in the argument-making skills of the experimental and control groups.

All of the exercises in the tutoring session and pre- and post-tests were based on program-generated Argument Contexts including some of those illustrated in the above figures. The pre- and post-tests, for instance, employed the Argument Contexts of Figures 1 and 5; students were asked to select the best case to cite for a side, employ it in an argument and to respond to the argument from the opponent's point of view. In all instances, students received textual descriptions of each of the cases involved in the various Argument Contexts, but they were not shown any of the graphic representations of the above figures. For each question, the grader simply assessed whether the student had taken advantage of the argument-making opportunities implicit in the Argument Context associated with the problem and assigned a grade on a five point scale.[4]

The results, reported in the footnote table, support the conclusion that program-generated Argument Contexts can be used to improve law students' argument-making performance, at least in a manual tutoring context. After the tutoring session, the members of the experimental group performed better than the control group. The pre-test results indicate that, initially, the experimental group members were slightly better than the control group, but, in light of the narrow spread of pre-test scores, that difference does not appear to have been substantial. After the tutoring session, the experimental group students peformed substantially better than those in the control group in both questions of the post-test. The performance difference was more pronounced for question 1 than question 2 of the post-test, an effect we attribute to the fact that question

[3]The advantages of integrating experimental evaluation with the initial phases of tutorial program design and implementation have been discussed in [Littman and Soloway1988].

[4]Since the criteria were objective, Kevin Ashley performed the grading; we are aware that an independent law professor, unconnected with this research effort, should grade the tests and are attempting to cajole one into cooperating. Here are the results:

	Control		Experimental			Maximal Score
Subject	1	2	3	4	5	
Pre-test	11	14	16	14	14	30
Post-test						
Q. 1	6	13	23	24	22	25
Q. 2	17	21	23	24	23	30

2 presented the most complex Argument Context the students had yet seen. See [Aleven and Ashley1992] for more details.

Discussion and Conclusions

To summarize, we have identified five important kinds of Dialectical Examples, standard configurations of cases which enable an arguer to justify rhetorical assertions effectively by example. Our computer program generates Argument Contexts, collections of cases that instantiate such Dialectical Example types, from an on-line database of cases according to a user's general specifications. We plan to incorporate the Argument Context generation program into a tutorial system to teach law students to argue with cases; it will provide a stock of instances of Dialectical Examples to teach novice advocates how to recognize, carry out and respond to the associated rhetorical moves. Although generating such examples is very hard for humans even when dealing with small numbers of cases, our program generates and organizes such examples quickly and effectively. In a preliminary experiment, we employed program-generated Argument Contexts manually to teach basic argument skills to first year law students with good results. Our ability to define such complex examples declaratively in terms of logical expressions of Loom concepts and relations affords two advantages: (1) the concept definitions are modular and relatively understandable (which makes them easier to read, modify, and explain) (2) complex queries can incorporate quantified variables (e.g., a query does not have to specify a current fact situation but can designate all responsive Argument Contexts involving any case as cfs – an impossible query for HYPO to perform).

References

Aleven, Vincent and Ashley, Kevin D. 1992. Automated Generation of Examples for a Tutorial in Case-Based Argumentation. In *Proceedings of the Second International Conference on Intelligent Tutoring Systems.*

Ashley, Kevin D. and Aleven, Vincent 1991. Toward an Intelligent Tutoring System for Teaching Law Students to Argue with Cases. In *Proceedings of the Third International Conference on Articial Intelligence and Law*, Oxford, England. 42–52.

Ashley, Kevin D. and Rissland, Edwina L. 1988. Waiting on Weighting: A Symbolic Least Commitment Approach. In *Proceedings AAAI–88.* American Association for Artificial Intelligence. St. Paul.

Ashley, Kevin D. 1989. Defining Salience in Case-Based Arguments. In *Proceedings IJCAI–89.* International Joint Conferences on Artificial Intelligence. Detroit.

Ashley, Kevin D. 1990. *Modeling Legal Argument: Reasoning with Cases and Hypotheticals.* MIT Press, Cambridge. Based on Ashley's 1987 PhD. Dissertation, University of Massachusetts, COINS Technical Report No. 88–01.

Ashley, Kevin D. 1991. Reasoning with Cases and Hypotheticals in HYPO. *International Journal of Man-Machine Studies.*

Clancey, W. J. 1983. The Epistemology of a Rule-based Expert System: A Framework for Explanation. *Artificial Intelligence* 20(3):215–251.

Collins, Allan and Stevens, Albert L. 1982. Goals and Strategies of Inquiry Teachers. In Glaser, Robert, editor 1982, *Advances in Instructional Psychology*, volume 2. Lawrence Erlbaum Associates, Hillsdale, NJ.

Gentner, D. 1983. Structure-Mapping: A Theoretical Framework for Analogy. *Cognitive Science* 7:155–170.

Littman, David and Soloway, Elliot 1988. Evaluating ITSs: The Cognitive Science Perspective. In Polson, Martha C. and Richardson, J. Jeffrey, editors 1988, *Foundations of Intelligent Tutoring Systems.* Lawrence Erlbaum Associates, Hillsdale, NJ.

MacGregor, Robert M. 1988. A Deductive Pattern Matcher. In *Proceedings AAAI–88*, Saint Paul, MN. American Association for Artificial Intelligence. 403–408.

McCarty, L. Thorne and Sridharan, N. S. 1981. The Representation of an Evolving System of Legal Concepts: II. Prototypes and Deformations. In *Proceedings IJCAI–81*, Vancouver, BC. International Joint Conferences on Artificial Intelligence.

McGuire, R.; Birnbaum, L.; and Flowers, M. 1981. Opportunistic Processing in Arguments. In *Proceedings IJCAI–81*, Vancouver, BC. International Joint Conferences on Artificial Intelligence. 58–60.

Rissland, Edwina L. and Skalak, David B. 1989. Combining Case-Based and Rule-Based Reasoning: A Heuristic Approach. In *Proceedings IJCAI–89.* International Joint Conferences on Artificial Intelligence. Detroit.

Rissland, Edwina L. and Soloway, E. M. 1980. Overview of an Example Generation System. In *Proceedings AAAI–80*, Stanford, CA. American Association for Artificial Intelligence.

Rissland, Edwina L.; Valcarce, E. M.; and Ashley, Kevin D. 1984. Explaining and Arguing with Examples. In *Proceedings AAAI–84*, Austin, TX. American Association for Artificial Intelligence.

Saunders, Kurt M. 1991. Knowledge acquisition session re legal methods instruction. Recorded transcript. Assistant Professor of Law, University of Pittsburgh School of Law.

Suthers, D. and Rissland, E. 1988. Constraint Manipulation for Example Generation. Technical Report 88-71, Department of Computer and Information Sciences, University of Massachusetts, Amherst, MA.

Winston, Patrick H. 1975. Learning Structural Descriptions from Examples. In Winston, Patrick H., editor 1975, *The Psychology of Computer Vision.* McGraw-Hill, New York.

Woods, William A. and Schmolze, James G. 1990. The KL-ONE Family. Technical Report TR-20-90, Center for Research in Computing Technology, Harvard University, Cambridge, MA.