A Computational Model of Argument for Legal Reasoning Support Systems

Thomas F. Gordon
Fraunhofer FOKUS
Kaiserin-Augusta-Allee 31
Berlin, Germany
thomas.gordon@fokus.fraunhofer.de

ABSTRACT

This paper presents a computational model of defeasible argument, including Walton's concept of argument schemes, for use in legal reasoning support systems. Building on ontologies from the Semantic Web, the model provides an integrating framework enabling diverse models of a variety of legal argumentation schemes, such as arguments from legislation, precedent cases and evidence, to be used together in a comprehensive system supporting argument construction, selection and evaluation, as well as the justification of legal decisions. Argument schemes in this model are interpreted as interactive, heuristic search procedures, to be used to help find and construct effective arguments during legal discourse. This contrasts with prior research, which uses argument schemes mainly to classify arguments post facto, during the reconstruction of arguments found in texts. In the model of argument schemes, we distinguish two kinds of critical questions, called presumptions and exceptions, depending on whether or not the condition expressed by the critical question is to be assumed to hold before the question has been asked. To support the evaluation of arguments, the model formally defines the presumptive validity of arguments and the acceptability of propositions, using proof standards.

Categories and Subject Descriptors

H.4.2 [[**Information System Applications**]: Types of Systems—*decision support*; J.1 [**Computer Applications**]: Administrative Data Processing—*law*

Keywords

Argumentation, Argument Schemes, Legal Reasoning Support Systems, Artificial Intelligence and Law, Semantic Web, Ontology Web Language (OWL)

1. INTRODUCTION

Great progress has been made in recent years in the field of Artificial Intelligence and Law in using formal or computational models to deeply analyze and understand ever more kinds of legal argument, including arguments from cases; rules from legislation; rationales; ethical or legal principals; values, purpose or policy; and evidence. Although some prior work has developed rich models capable of handling several kinds of argument, in particular arguments from cases and rules [3, 21] , there is as yet no generally accepted open framework enabling diverse models of all kinds of legal argument to be used together, so as to facilitate the development of a comprehensive marketplace of tools and components for supporting legal reasoning in an integrated way.

The thesis of this paper is that Walton's theory of argument schemes [28], suitably interpreted and refined, can provide the foundation for such an open framework. The paper presents a mostly functional computational model of Walton's theory and the rationale for this model in terms of identified requirements and use cases.

The rest of this paper is organized as follows: Section 2 describes the goals and requirements to be met by the model, including the range of legal reasoning tasks to be supported; this is followed in Section 3 with our model of domains of discourse, i.e. the model of the things about which arguments are about; Section 4 explains the model of issues, arguments, and cases; Section 5 presents the model of argument schemes as heuristic search procedures; Section 6 presents the remaining parts of the model, in the context of the various reasoning tasks to be supported; Section 7 discusses related prior work; and Section 8 closes the paper with conclusions and some ideas for future work.

2. GOALS AND REQUIREMENTS

The main requirement for the model is that it should be suitable as an integrating framework for a wide variety of legal reasoning support systems, for any kind of legal argument. In addition, the model should be useful not only for the post factor reconstruction, evaluation or visualization of arguments in legal texts, such as published court decisions, but also for legal knowledge systems which help people to construct arguments as they participate in discourses about their legal rights and obligations and for helping judges and other third parties to construct justifications or rationales for their legal decisions. That is, our goals are not only analytical, to better understand legal argument, but also have an engineering purpose, to be suitable for use in practical software for supporting actual legal work. Figure 1 illus-

trates the use cases we want to support.

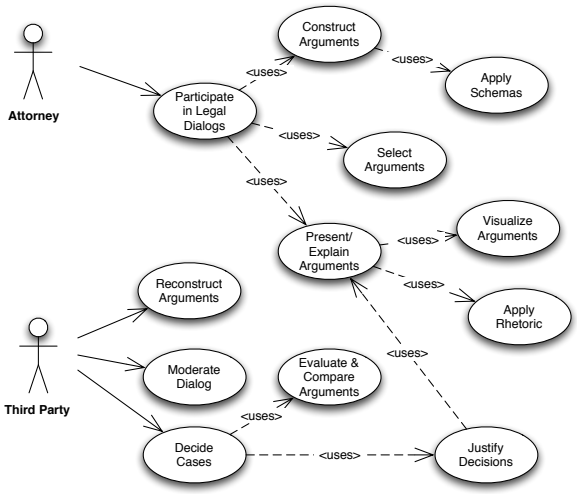

Figure 1: Legal Argumentation Uses Cases

These requirements have corollaries: the model should be a precise specification of data structures and algorithms, sufficient for implementation in computer programs and, just as importantly, re-implementation by others. This is what we mean by computational model: a high-level specification of computer software.

The model of argument should be as simple as possible (c.f. Occam's Razor), but unlike for example Dung's model [8], not so simple that features of legal argument critical for practical legal reasoning support systems are abstracted away.

Finally, we would like our model to be compatible with relevant standards, in particular those emerging in the context of the so-called Semantic Web [1], such as the Resource Description Framework [16] or the OWL Web Ontology Language [17].

3. DOMAINS OF DISCOURSE

Arguments can be about many things. They can be about whether some proposition is logically entailed by some set of premises, or whether a set of propositions is consistent. Indeed, much work on argumentation has been restricted to just these kinds of logical issues. But there are other types of arguments as well. For example, one can argue about whether some object exists (e.g. the Loch Ness monster) or whether two objects have a particular relationship to one another (e.g. whether some man is the father or a child, in a paternity suit). One is tempted to call these factual issues, but legal issues can be viewed this way as well, for example, whether some legal norm is expressed in some legal text, such as a statute or precedent case. Finally, arguments may also be about terminological issues, such as whether or not one concept is subsumed by another (e.g. whether humans are animals). Of particular interest in legal reasoning are disputes about whether some nonlegal concept (e.g. skateboards) should be subsumed by some open-textured legal concept (e.g. vehicles, in the technical sense of some statute prohibiting vehicles from parks).

We adopt the model of the Semantic Web for representing domains of discourse. Essentially, these are directed, labeled graphs, where the nodes represent objects, entities or concepts and the arcs represent binary relationships. The Resource Description Framework (RDF) of the Semantic Web is a concrete, XML syntax for such graphs. Graphs of this kind have a long tradition in the Artificial Intelligence field of knowledge representation (e.g. semantic nets, conceptual graphs). One advantage is that they can be easily visualized. The Ontology Web Language (OWL) uses RDF to represent terminological knowledge.

More concretely, we model domains as an `atom set`, where atoms are similar to RDF tuples:[1]

```
type atom =
{ entity: id, attribute: id,  value: datum }
```

The entity and attribute identifiers could be Universal Resource Identifiers (URIs), as in RDF. The value may any datum; i.e. an identifier, number, string, date, or, recursively, another atom. The model does not specify a concrete syntax for atoms, but for the sake of tradition we will use Lisp s-expressions in this paper. For example, here are some atoms:

```
(subclass skateboard vehicle)
(forbidden vehicle park)
(isa sb skateboard)
(asserts gloria (killed joe sam))
```

Figure 2 shows one way to visualize such domain models.

From a logical perspective atoms can be interpreted as ground atomic propositions, where attributes are binary predicates. Notice, however, that as the value of an atom can represent another atom, propositions can be higher-order. This model of discourse domains is expressive enough, we claim, to handle all factual and legal issues of practical relevance.

4. ISSUES, ARGUMENTS AND CASES

Legal argumentation here is viewed as a model construction process [9], about which atoms to include in the domain model. An `issue` is a record for keeping track of the arguments pro and con each proposed or claimed value of some attribute of an entity, the status of each value in the discourse (accepted, rejected, undecided), and the proof standard applicable to this issue.

```
type issue =
{ id: id,
  entity: id,
```

[1]We use Standard ML [14], with some minor ad hoc extensions, as a meta-language in this paper. The model can be implemented in any high-level programming language. We are using Scheme [5] to implement a prototype.

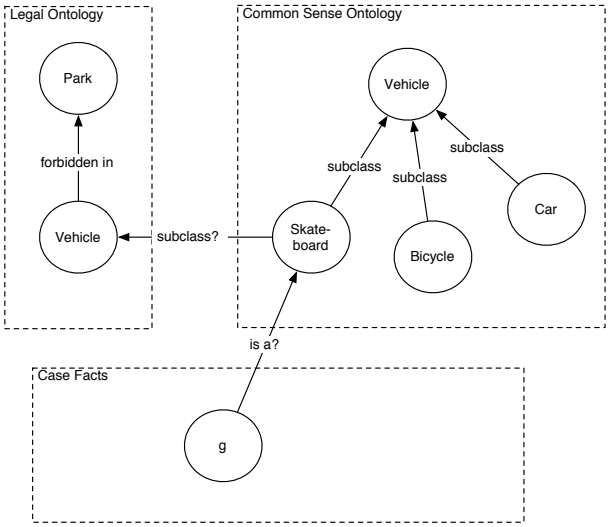

Figure 2: Example Domain Model

```
  attribute: id,
  standard: proof-standard,
  position: position list }

type position =
{ status: {accepted, rejected, undecided},
  value: datum,
  pro: argument list,
  con: argument list }
```

To keep track of commitments, this model would need to be extended with information about who claimed each value and asserted each argument.

Arguments record applications of argument schemes. More precisely, an `argument` in this model is a record of the following type

```
type argument =
{ id: id,
  direction: {pro, con},
  scheme: id,
  consequent: atom,
  antecedent: atom list,
  presumptions: atom list,
  exceptions: atom list }
```

This assumes that each argument scheme has been given a unique identifier. Notice that the model distinguishes three types of premises, called antecedents, presumptions and exceptions. As will be discussed more throughly in Section 6.1, the model of presumptive validity of arguments handles each kind of premise differently. Presumptions are assumed to be acceptable; exceptions are assumed not acceptable; and antecedents must have sufficient support, depending on the applicable proof standard, to be deemed acceptable.

Presumptions and exceptions are a refinement of Walton's concept of critical questions. Critical questions enumerate specific ways to defeat arguments matching some argument scheme. But so long as an issue has not been raised by actually asking some critical question, we would like to be able to express which answer to presume, particularly since one of the main goals of such a model of argumentation is to provide an adequate account of presumptive or defeasible reasoning. The distinction between presumptions and exceptions here provides this ability.

Here is an example argument, applying the argument from expert opinion scheme:

```
id: arg-1
direction: pro,
scheme: argument-from-expert-opinion,
consequent: (mentally-ill defendant true),
antecedents:
   (isa e expert-testimony)
   (domain e d)
   (assertion e (mentally-ill defendant true))
   (within d (mentally-ill defendant true)),
presumptions:
   (credible e true)
   (based-on-evidence e true),
exceptions:
  (trustworthy e false)
  (consistent-with-other-experts e false)
```

This model of argument is rich enough to handle all kinds of defeat relationships among arguments, as required for defeasible legal reasoning. Rebuttals can be modeled as arguments in the opposite direction for the same consequent. (If the first argument is `pro` the consequent, the rebuttal would be an argument `con` the same consequent, and vice versa.) Premise defeat can be modeled with arguments `con` an antecedent or presumption, or `pro` an exception. Undercutting defeaters are a bit trickier. The idea of an undercutting defeater is to argue against the rule or warrant which was applied to create the argument. But the model of argument here is not restricted to arguments from rules. Such arguments are just a special case, resulting from the application of some argument scheme for reasoning with rules. To handle undercutting defeaters, we assume that such argument schemes will include suitable presumptions or exceptions in the arguments generated, for example to allow an issue to be made out of the applicability of a rule.

Finally, a `case` is modeled here simply as a set of issues. Because of the structure of issues, this case model provides us with not only all of the issues raised in the case (thus far), but also all of the positions and arguments asserted for each issue. The `findings` of the case are the accepted and rejected positions of each of the issues in the case. The accepted positions induce a domain model and thus can be visualized as a semantic net, as discussed and illustrated previously. This illustrates one way our computational model can be understood as adhering to the model construction conception of legal reasoning.

5. ARGUMENT SCHEMES

Argument schemes are "forms of argument" for "stereotypical patterns of human reasoning" [2]. They can be viewed as

inference rules for presumptive reasoning. Arguably, argument schemes are a further elaboration of Toulmin's warrant concept [25]. Argument schemes have been used primarily to classify, post facto, arguments in natural language texts. For this purpose, the schemes are used as patterns to support the *reconstruction* of the form of an argument during the interpretation of the text. This is in stark contrast to the usual function of inference rules, as tools for deriving new conclusions from premises.

Our aim here is to due justice to the inference rule conception of argument schemes, by modeling them in such a way that they can be used as tools to find, construct or generate arguments. One way to approach this task is to try to define some kind of pattern language for schemes and then design some kind of pattern matcher for applying these patterns to data to generate arguments. Although this idea may be worth pursuing, in future research, our current approach is more abstract and somewhat less ambitious. The computational models for the many kinds of legal reasoning are so diverse (e.g. arguments from legislation, precedent cases, legal principles, values and evidence) that it would be quite a challenge to design a single pattern language which is not only general enough to match the data used in these models, but also be sufficiently open to cover future models. Instead, we have chosen the following simple and abstract model of argument schemes:

```
type scheme =
{ id: id,
  find: atom * {pro, con} * case ->
          argument stream }
```

The id of a scheme is to be referenced in the arguments generated by the scheme. The actual work is done by the `find` function. Its task is to search for arguments pro or con some goal atom. A case record of the discourse thus far is provided in the third parameter, to enable the procedure to take all available information into account in its heuristic strategy for finding or prioritizing arguments. The procedure returns a stream of arguments. A stream is a sequence whose members can be generated on demand, as they are accessed, rather than all at once. This generalization allows implementations of particular argument schemes a great deal of flexibility. For example, a rule-based system could be used to implement a scheme, and backtrack to find alternative arguments.

One might argue that this model of schemes does not provide any assistance in modeling or implementing particular arguments schemes, for example to reason with legislation or precedent cases. But that would be asking too much. Our aim here is to provide a conventional way to pass information about the goal and state of the discourse into legal reasoning components implementing various schemes and to obtain in return arguments in a standard format, to enable these components to be used together, in an integrated way, to provide comprehensive support for many kinds of argumentation tasks. We will have succeeded if it is practically possible to wrap existing legal reasoning support systems with a programmer's interface matching our type signature for argumentation schemes and this can be shown to provide an effective platform for integrating diverse reasoning services.

We do not assume or require implementations of arguments schemes to be completely automatic. On the contrary, we expect many schemes to be implemented as interactive tools which assist human users in finding arguments. The scheme for arguments from expert opinion, for example, might be implemented as an interactive web service which helps users to find and interview domain experts.

6. LEGAL REASONING TASKS SUPPORTED

As illustrated in Figure 1, there are a number of use cases of legal argumentation that our model is intended to to support. In Section 5, we discussed how our model of argument schemes has been designed so as to support the construction of arguments. In this section we will explain how the model supports additional use cases, in particular evaluating arguments to determine their presumptive validity, selecting arguments from an adversarial viewpoint and, finally, justifying legal decisions.

6.1 Argument Evaluation

By argument evaluation we mean deciding whether an argument is presumptively `valid`, considering all the arguments which have been asserted thus far in the discourse. The definition of validity we present here is recursive: validity is defined in terms of the `acceptability` of the premises of the argument; if an issue has been raised about some premise, acceptability depends on the `proof standard` associated with the issue; finally, proof standards are defined in terms of the validity of the competing arguments, pro and contra, each position of an issue. Thus, no matter where we begin, it will be necessary to make forward references to concepts defined later. We begin with presumptive validity (for the sake of brevity to be called simply validity hereafter). An argument is `valid` iff:

- all of its antecedents are `acceptable`;
- all of its presumptions `at-issue` are `acceptable`; and
- none of its exceptions `at-issue` are `acceptable`.

Notice that according to this definition presumptions *not* at issue are in effect presumed to be acceptable, as their name suggests. Conversely, exceptions *not* at issue are presumed in effect not to be acceptable. Notice also that these presumptions are context sensitive: the same atom may in principle be a presumption of one argument and an exception in another. So long as the atom is not at issue, it can be presumed to be both acceptable and not acceptable at the same time, in the contexts of separate arguments. This is because argument schemes control presumptions and different schemes may implement different policies.

Recall that antecedents, presumptions and exceptions are all `atoms`. An atom is `at-issue` iff there exists an `issue` in the case such that:

- the `entity` of the issue is the same as the `entity` of the atom; and
- the `attribute` of the issue is the same as the `attribute` of the atom.

Notice that the `value` field of the atom does not play a role in this definition. It does not matter whether the particular value of the atom has been proposed or asserted as a value in the issue. In this conception of issue, the question is not so much whether or not a particular value is acceptable, but rather *which* values to accept for a particular attribute of a particular entity.

Perhaps this is a good place to mention that ontologies can place restrictions on the number of values which can be accepted. Recall that entities and attributes can be ids which refer to concepts and relations in OWL ontologies, using URIs. How to handle these number restrictions in this argumentation framework is an open question. Perhaps the model of proof standards, below, should be somewhat extended to take these constraints into account, but I do not yet see how the proof standard could select among the alternatives if too many values are acceptable. It might appear that an argument scheme for number restrictions from ontologies could be realized, but here too it is not clear how such a scheme could choose among the competing values. The alternative I prefer at the moment is to check number constraints using an OWL reasoner in a post processing phase. Argumentation would be used to generate models, and other tools such as OWL reasoners could then be used to further evaluate or check these models.

Let us now turn to the definition of the acceptability of atoms. Acceptability is a function with this signature:

```
val acceptable: atom * case ->
                {undefined, true, false}
```

and is defined as follows:

```
acceptable (atom-1, case-1) =
```

- `undefined`, if `atom-1` is not at issue in `case-1`;
- `true`, if `atom-1` is at issue in `case-1` and its value is the value of one of the positions admitted by the proof standard of the issue;
- `false`, otherwise.

Don't be confused by the first clause of this definition, which leaves the acceptability of an atom undefined if it is not at issue. This may seem to conflict with the definition of argument validity, which we explained as presuming, "in effect", that presumptions and exceptions are acceptable or, respectively, not acceptable, so long as they are not at issue. The definition of acceptability here does not depend on the argument context.

The final element relevant for determining argument validity is the model of proof standards. A proof standard is modeled as a function of the following type:

```
type proof-standard = issue * case -> datum list
```

Recall that issues are assigned a proof standard. The idea is to apply the proof standard to this issue to determine which of the proposed values, in the positions of the issue, meet the proof standard.

Here are three definitions of this type for common legal proof standards, to help validate the adequacy of the model:

Scintilla of Evidence. A position meets this standard iff at least one `valid` argument exists `pro` the position.

Preponderance of the Evidence. A position meets this standard iff there are more `valid` arguments `pro` this position than `valid` arguments `con` this position.

Beyond a Reasonable Doubt. A position meets this standard iff all of its `pro` arguments and none of its `con` arguments are valid.

None of these proof standards weigh, order or prioritize arguments. They simply count, in various ways, how many arguments pro and contra some position are valid. In principle, however, proof standards can be implemented which make use of some kind of weighing function to order arguments. We would need to experiment with particular proposals for weighing functions in order to evaluate whether the current model contains the information required by weighing functions, at the right level of abstraction.

One can also imagine proof standards which support argumentation about the priority of arguments [12, 20, 13] within the same framework. Arguments and issues are *reified* in this model; they may be referenced in both the entity and value fields of atoms by their ids. This makes it possible to argue about arguments: issues about arguments can be raised and both the premises and conclusions of arguments can be about arguments. For example, the principle of *lex superior* (norms from a higher level authority have priority over norms from a lower level authority) could be modeled as an argument scheme which generates arguments like this:

```
id: arg-1
scheme: lex-superior
consequent: (prior arg-1 arg-2)
antecendents:
   (rule arg-1 rule-1)
   (authority rule-1 federal-law)
   (rule arg-2 rule-2)
   (authority rule-2 state-law)
   (superior federal-law state-law)
```

We have not done a thorough analysis of the computational properties of argument evaluation in this model. Since cases are finite data structures, consisting of a finite number of issues and arguments, and since both the premises and conclusions of arguments are ground atomic propositions, it seems reasonable to consider whether the presumptive validity of arguments and the acceptability of atoms is decidable.

There are at least two problems, however. First, validity and acceptability are defined in terms of proof standards which have been left open to be defined in applications. Second, a straightforward implementation of the recursive definitions of validity and acceptability will not terminate in the face of cyclic arguments. The semantics of cases with cyclic arguments is not entirely clear. If necessary, the definitions of validity and acceptability could be restricted to "well-formed", cycle-free cases. Although from a theoretical perspective these may be important problems, from a practical point of view these issues can be overcome. Models of proof standards need to be evaluated separately before being accepted for use. And cycles may be avoidable using an argumentation protocol which prohibits arguments which would introduce cycles from being asserted.

6.2 Advocacy

Given the arguments exchanged thus far in some proceeding, which arguments should an attorney or other advocate make on behalf of his client next? This task can be supported using our model of argumentation by following this procedure:

1. Create a private copy of the case.
2. Search for new arguments by applying argument schemes and update the private copy of the case with the arguments generated. Repeat this step until the client's position is acceptable.
3. Select a *minimal* subset of the new arguments such that the client's position is acceptable given the prior arguments and the selected new arguments.
4. Assert the selected subset of the new arguments into the shared, public copy of the case.

This is the basic procedure. One can imagine more elaborate alternatives, such as searching also for arguments for the other side and choosing to assert a more complex set of arguments, rather than a minimal set, taking anticipated counter-arguments into account without waiting for the other side to first make them. Such elaborate moves may make sense from a rhetorical perspective, even if they are unnecessary from a strictly "logical" point of view.

The selection of a minimal subset of arguments supporting a position can be viewed as a kind of *abduction* problem, where acceptability is used instead of logical consequence to test whether the goal proposition is supported by the selected hypotheses.

6.3 Decision Justification

The decision about which values of some attribute to accept need not be determined by the arguments asserted. More than one value may be acceptable. Depending on the ontologies used, multiple values may be accepted without being inconsistent. Conversely, it may be necessary to select among the acceptable values in order to construct a model which satisfies all the constraints of the applicable ontologies.

Thus, the kind of defeasible reasoning supported by this model of argumentation is "credulous", rather than "skeptical". It constrains but does not determine decisions. In the legal context, it leaves judges and other kinds of arbiters considerable room for discretion.

Various procedures for justifying decisions are conceivable, depending for example on whether the judge is permitted to make additional arguments of his own or whether he must construct the justification of his decision solely from the arguments made by the parties. Supposing the latter is the case, here is a goal-directed procedure which enables a judge to select among alternative acceptable conclusions and produce a justification in terms of the available arguments:

1. Decide the main issues of the case, by selecting among the acceptable positions of these issues. These decisions can be noted by setting the status of each selected position to `accepted`.
2. Select a *maximal* subset of the arguments in the case such:
   - The accepted positions remain acceptable; and
   - The resulting model satisfies the constraints of all the ontologies used in the model.
3. If no such subset of the arguments exists, the problem may have been overconstrained in the first step, by accepting too many positions. In this case, backtrack to the first step and retract the decision to accept one or more positions, by changing their status back to `undecided`.
4. To note the final decisions, compare the resulting case with the initial version, the one containing all the arguments made by the parties, set the status of all positions not acceptable in the resulting case to `rejected` in the initial version, and the status of all acceptable positions in the resulting case to `accepted` in the initial version.

As for advocacy, the task of justifying decisions can be viewed as a kind of abduction problem. The arguments serve the role of the hypotheses; the desired decisions the role of the "evidence" to be "explained" by a subset of the hypotheses supporting the evidence and meeting additional constraints. Again, the support relation here is presumptive validity rather than classical entailment. The additional constraints we have formulated are different for the justification of decisions than for the selection of the next arguments to make by advocates. Whereas we have required the judge to retain as many of the arguments made by the parties as possible, we have allowed advocates to reveal only as many arguments to the other side as necessary to make their point. Moreover, we haven't required advocates to check their arguments against the constraints from applicable ontologies, preferring to leave this checking task up to the judge during the construction of his decision and its justification.

7. RELATED WORK

The idea of developing a computer model for managing support and justification relationships between propositions

goes back to research on "truth" or reason maintenance systems in Artificial Intelligence, beginning with Jon Doyle's Truth Maintenance System [7]. Probably the most famous system of this kind is Johann DeKleer's Assumption-Based Truth Maintenance System [15]. The system presented here is more like Doyle's original system; the way it supports defeasible reasoning using arguments with different kinds of premises is reminiscent of Doyle's use of `in` and `out` lists in justifications.

The author's prior work on the Pleadings Game [12] included a formal model of dialectical graphs, for recording various kinds of support and defeat relationships among arguments. Arguments were modeled quite differently in the Pleadings Game, as a set of formulas in a logical language, based on Conditional Entailment [11]. Whereas support and defeat relationships were implicit and required deep logical reasoning in the Pleadings Game to recognize, they are readily apparent from the structure of issues and arguments in the system presented here. Similarly, whereas issues in the Pleadings Game were derived from a deep analysis of the relevance of propositions for proving the main issue of the case, they are directly raised and asserted by users in the present model. Finally, a knowledge representation language for legal rules and a method for formulating arguments using these rules was "hard-wired" into the Pleadings Game. The current model of argumentation is designed to be an open integration framework for various kinds of legal argumentation schemes, using whatever kind of knowledge representation is appropriate for each kind of scheme.

Building on prior work by Brewka and Gordon [4], the concept of an *argumentation framework* was introduced by Henry Prakken [18] as part of a three-layered model for dialectical systems, consisting of a logic layer for representing knowledge and deriving arguments; the argumentation framework for recording and evaluating arguments, using support and defeat relationships; and finally a protocol layer defining the speech acts for making assertions, asking questions and the like. He later added a fourth, "strategic" layer for planning argument moves.

The Zeno Argumentation Framework [13], by the author and Niklaus Karacapilidis, was an argumentation framework based on an extended version of Horst Rittel's Issue-Based Information System (IBIS) model of argumentation [23]. It was intended to be simple enough for use in web-based mediation systems and targeted to support practical decisions in what Douglas Walton in The New Dialectic [30] calls deliberation dialogs, about what action to take. Like the present work it did however include a model of proof standards, supported arguments about preferences among arguments and was intended as an integrating framework for arguments from "heterogeneous information sources and models". But unlike the current work, Zeno did not include a model of the concept of argument schemes or make use of shared ontologies to provide a kind of semantic glue between such heterogeneous models.

Henry Prakken, Chris Reed and Douglas Walton were the first to make use of argumentation schemes in AI models of legal argument [19], but this work was focussed on arguments about evidence. Bart Verheij used argument schemes in a methodology for modeling legal knowledge as rules in his DefLog formalism [26]. Verheij's use of schemes in a methodology for repesentation knowledge in a single formalism thus has a different purpose and scope than our interpretation of schemes as a type of function for integrating and encapsulating diverse formalisms for modeling of legal knowledge. Perhaps for this reason, Verheij's work did not address the question of how to make the premises and conclusions of arguments generated from diverse reasoning components comparable, as we have tried to do with ontologies here. Verheij's work in [26] was however the source of inspiration for distinguishing between different kinds of critical questions, which we have called presumptions and exceptions.

Araucaria [24] is a computer program for helping users to reconstruct arguments in natural language texts and visualize them in a diagram. Premises and conclusions identified by the user in the text can be added to the diagram by simply marking up parts of the text. Arguments are added by drawing arrows between nodes in a diagram. Argument schemes can be used to classify arguments. The diagramming method used is described by Walton in [29]. Araucaria includes an XML format, called the Argument Markup Language (AML), for storing and exchanging argument diagrams. Unfortunately, the model of argument underlying the argument diagramming method support by Araucaria is quite different than the model presented here. Thus, Araucaria and AML cannot be used to visualize or represent the arguments constructed using this model.

For a more thorough treatment of prior related work on computational models of argument for use in support systems than can be provided here, the reader is referred to two recent monographs: Argumentation Machines [22], edited by Chris Reed and Timothy Norman, and Virtual Arguments, [27], by Bart Verheij.

8. CONCLUSIONS AND FUTURE WORK

This paper has presented a computational model of defeasible argument, including Walton's concept of argument schemes, for use in legal reasoning support systems. Building on ontologies from the Semantic Web, the model provides an integrating framework enabling diverse models of a variety of legal argumentation schemes, such as arguments from legislation, precedent cases and evidence to be used together. We have sketched how the model may be used to support argument construction and evaluation, argument selection by advocates, and the justification of legal decisions by judges or other third parties. Argument schemes in this model are interpreted as interactive, heuristic search procedures, to be used to help find and construct effective arguments during legal discourse. This contrasts with prior research, which uses argument schemes mainly to classify arguments post facto, during the reconstruction of arguments found in texts. In the model of argument schemes, we have distinguished two kinds of critical questions, called presumptions and exceptions, depending on whether or not the condition expressed by the critical question is to be assumed to hold before the question has been asked. To support the evaluation of arguments, the model includes formal definitions of the presumptive validity of arguments and the acceptability of propositions, in terms of proof standards.

As appropriate for a workshop, the model presented here is work in progress, at an early stage of development. A prototype implementation, in Scheme, is near completion, but the model has yet to be tested or validated using this implementation with realistic data. For validation purposes, it will be necessary to implement some legal argument schemes, either by trying to wrap existing legal reasoning support systems or by implementing prototype systems especially for this purpose. Whichever route we choose, we will want to be careful to cover a wide range of legal argument schemes, including at least arguments from legislation (rules) and from precedent cases.

There are some aspects of the current model with which we are not entirely content. In particular, the necessity in the current model of checking constraints from ontologies in a kind of post-processing phase seems inelegant and rather complex. We would prefer to use ontologies as just another source of arguments, using arguments schemes to encapsulate ontological reasoners. The problem is that some of the constraints, in particular number constraints, restrict the set of values of some attribute without providing any arguments about which values to prefer.

Another problem for future research is how to produce a clear and understandable presentation of the justification of a decision. The necessary arguments are all contained in the case model, but how can they be presented and, ideally, visualized in a clear way, given the links between arguments, ontologies and domain models and the open architecture for arbitrary proof standards. None of the existing diagramming techniques for arguments [25, 6, 29] directly apply. Perhaps, as in the Unified Modeling Language [10], it will be necessary to use several kinds of diagrams for different perspectives on the model.

9. ACKNOWLEDGEMENTS

I would like to thank Wolfgang Bibel, Trevor Bench-Capon, Doug Walton, and several of my colleagues at Fraunhofer FOKUS, Dirk Arendt, Olivier Glassey and Jonas Pattberg, for fruitful discussions on the subject of this paper.

10. REFERENCES

[1] T. Berners-Lee, J. Hendler, and O. Lassila. The semantic web. *Scientific American*, 284(5):34–43, May 2001.

[2] F. Bex, H. Prakken, C. Reed, and D. Walton. Towards a formal account of reasoning with evidence: Argumentation schemes and generalizations. *Artificial Intelligence and Law*, 11(2-3), 2003.

[3] L. K. Branting. *Reasoning with Rules and Precedents: A Computational Model of Legal Analysis*. Kluwer Academic Publishers, Dordrecht, 1999. Book version of 1991 PhD Thesis.

[4] G. Brewka and T. F. Gordon. How to buy a porsche: An approach to defeasible decision making. In *Working Notes of the AAAI-94 Workshop on Computational Dialectics*, pages 28–38. Seattle, Washington, 1994.

[5] W. Clinger and J. Rees. Revised report on the algorithmic language Scheme. November, 1991.

[6] J. Conklin and M. Begeman. gibis: A hypertext tool for exploratory policy discussion. *ACM Transactions on Office Information Systems*, 6(4):303–331, 1988.

[7] J. Doyle. A truth maintenance system. *Artificial Intelligence*, 12:231–272, 1979.

[8] P. M. Dung. On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games. *Artificial Intelligence*, 77(2):321–357, 1995.

[9] H. Fiedler. Expert systems as a tool for drafting legal decisions. In A. A. Martino and F. S. Natali, editors, *Logica, Informatica, Diritto*, pages 265–274. Florence, 1985.

[10] M. Fowler and K. Scott. *UML Distilled — A Brief Guide to the Standard Object Modeling Language*. Addison Wesley Longman, Inc., 2nd edition, 2000.

[11] H. Geffner and J. Pearl. Conditional entailment: Bridging two approaches to default reasoning. *Artificial Intelligence*, 53(2-3):209–244, 1992.

[12] T. F. Gordon. *The Pleadings Game; An Artificial Intelligence Model of Procedural Justice*. Kluwer, Dordrecht, 1995.

[13] T. F. Gordon and N. Karacapilidis. The Zeno argumentation framework. In *Proceedings of the Sixth International Conference on Artificial Intelligence and Law*, pages 10–18. Melbourne, Australia, 1997.

[14] R. Harper, D. MacQueen, and R. Miller. Standard ML. March(ECS-LFCS-86-2), 1986.

[15] J. d. Kleer. An assumption-based tms. *Artificial Intelligence*, 28, 1986.

[16] F. Manola and E. Miller. RDF primer, 2004.

[17] D. L. McGuinness and F. van Harmelen. OWL web ontology language overview, 2004.

[18] H. Prakken. From logic to dialectic in legal argument. In *Proceedings of the Fifth International Conference on Artificial Intelligence and Law*, pages 165–174. Maryland, 1995.

[19] H. Prakken, C. Reed, and D. Walton. Argumentation schemes and generalisations about evidence. In *International Conference on Artificial Intelligence and Law*, pages 32–41, Edinburgh, 2003.

[20] H. Prakken and G. Sartor. A dialectical model of assessing conflicting argument in legal reasoning. *Artificial Intelligence and Law*, 4(3-4):331–368, 1996.

[21] H. Prakken and G. Sartor. Modelling reasoning with precedents in a formal dialogue game. *Artificial Intelligence and Law*, 6(2-4):231–287, 1998.

[22] C. Reed and T. J. Norman, editors. *Argumentation Machines — New Frontiers in Argument and Computation*. Kluwer Academic Publishers, Boston/Dordrecht/London, 2003.

[23] H. W. Rittel and M. M. Webber. Dilemmas in a general theory of planning. *Policy Science*, 4:155–169, 1973.

[24] G. W. A. Rowe, C. A. Reed, and J. Katzav. Araucaria: Marking up argument. In *European Conference on Computing and Philosophy*, Glasgow, 2003.

[25] S. E. Toulmin. The uses of argument. 1958.

[26] B. Verheij. Dialectical argumentation with argumentation schemes: An approach to legal logic. *Artificial Intelligence and Law*, 11(2-3):167–195, 2003.

[27] B. Verheij. *Virtual Arguments*. TMC Asser Press, The Hague, 2005.

[28] D. Walton. *Argument Schemes for Presumptive Reasoning*. Erlbaum, 1996.

[29] D. N. Walton. *Argument structure : a pragmatic theory*. Toronto studies in philosophy. University of Toronto Press, Toronto ; Buffalo, 1996. Douglas Walton. ill. ; 24 cm.

[30] D. N. Walton. *The new dialectic : conversational contexts of argument*. University of Toronto Press, Toronto; Buffalo, 1998. 24 cm.