An Overview of the Carneades Argumentation Support System

Thomas F. Gordon
Fraunhofer FOKUS
Berlin, Germany
thomas.gordon@fokus.fraunhofer.de

Abstract

Carneades is both a mathematical model of argumentation and a software toolbox providing support for argument evaluation, construction and visualization. Here we present an overview of the current version of the Carneades toolbox, explaining how the tools can be used to support argumentation tasks and providing some technical information about how they have been implemented.

1 Introduction

Carneades is a set of Open Source software tools for supporting a range of argumentation tasks, based on a mathematical model of Doug Walton's philosophy of argumentation and developed in collaboration with him over the course of several years, beginning in 2006.[1] Work on Carneades is a research vehicle for studying argumentation from a more formal, computational perspective than is typical in the field of informal logic, and for developing prototypes of tools designed to be useful for supporting real-world argumentation in practice. Thus there have been several versions of Carneades, as we experiment with different formal models of various argumentation tasks and with different ideas for tools which might be useful for helping people to argue more effectively. Carneades is work in progress. There is still much to do and feedback from an empirical evaluation of Carneades tools may lead to further changes in the system.

We began this project by doing a use-case analysis of common argumentation tasks, as illustrated in Figure 1.[2] This is another three-layered model of argumentation tasks (Brewka and Gordon, 1994; Prakken, 1995), where the three layers are inspired by Aristotle's distinctions between logic, dialectic and rhetoric. We do not claim, however, that our conceptualization of these three

[1] http://carneades.berlios.de

[2] See also (Macintosh et al., 2009), which applies this use-case diagram in a survey of argumentation software, to classify avaiable tools.

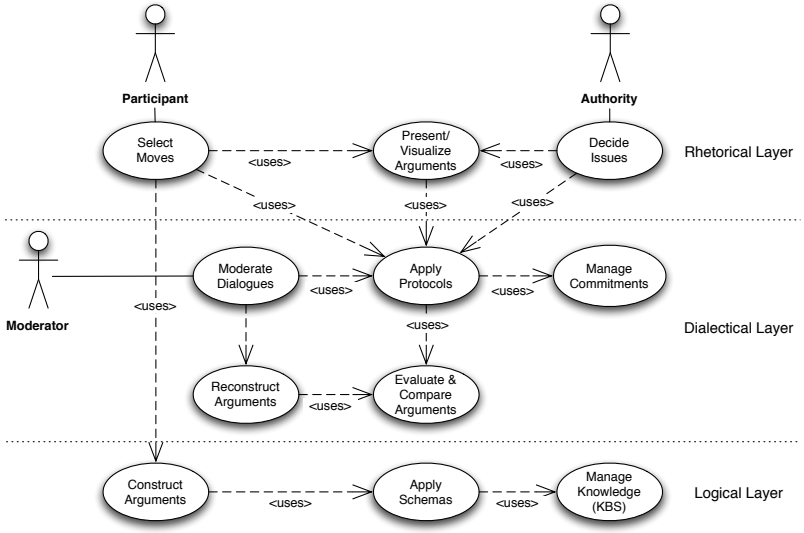

Figure 1: Argumentation Uses Cases

layers is perfectly faithful to Aristotle's views on these topics. Indeed, we anticipate that there may be some disagreement about how the tasks have been distributed among the layers in this diagram. For us, the important thing is not so much whether a particular task is considered logical, dialectical or rhetorical, as the simple recognition of the task as being important for the performance of some role in argumentation dialogues, and to try to capture which tasks depend on the results of which other tasks.

The logical layer, at the bottom of the diagram, covers the constructon of arguments from data, information, models and knowledge. We intend the sources of arguments to be very broad, ranging from sensory data, witness testimony and others kinds of evidence, across arguments from the interpretation of natural language texts, up to purely formal derivations of arguments from propositions expressed in some formal language, such as predicate calculus. We view argumentation schemes (Walton et al., 2008) not only as a useful tool for reconstructing and evaluating past arguments in natural language texts, but also as templates helping to guide users as they construct, 'invent' or generate their own arguments to put forward in ongoing dialogues (Gordon and Walton, 2009b).

The dialectical layer, in the middle of the diagram, covers tasks relevant for comparing and aggregating potentially conflicting or competing arguments, put forward by opposing parties in argumentation dialogues, such as legal procedures before courts. Procedural rules, often called 'protocols', regulate the allocation the burden of proof among the parties, the assignment of proof standards to issues, resource limits, such as due dates for replying or limiting the number

of turns which may be taken, and criteria for terminating the process, among other matters.

Finally, the rhetorical level, at the top of the diagram, consists of tasks for participating effectively in argumentation dialogues, taking into consideration the knowledge, experience, temperament, values, preferences and other characteristics of audiences, in particular one's opponent in a dispute. However, rhetoric is not only concerned with methods for taking advantage of an opponent to win a dispute. It is also about expressing arguments in clear ways which promote understanding, given the needs of the audience. We include at this level techniques for visualizing sets of interrelated arguments, i.e. *argument graphs*, as a particular class of methods for presenting arguments in ways which promote understanding.

Notice that the application scenarios which interest us, and which we want to support with software tools, are centered around dialogues, typically with two or more parties, in which claims are made and competing arguments are put forward to support or attack these claims. Following Walton, we recognize that there are many kinds of dialogues, with different purposes and subject to different protocols (Walton, 1998). This focus needs to be contrasted with the mainstream, relational conception of argument in the field of computational models of argument, typified by Argumentation Frameworks (Dung, 1995), which views argumentation not as a dialogical process for making justified decisions which resolve disputed claims in the face of resource limitations, but as a method for inferring consequences from an inconsistent set of propositions, by maximizing in some way, depending on the semantics of the particular approach, the number of propositions in the set which can be assumed to be true simultaneously. To see the difference between these conceptions of argument, notice that a proposition which has not been attacked is acceptable in this relational model of argument, in all common semantics, whereas in most dialogues, in particular persuasion dialogues, a proposition which has not been supported by some argument is typically not acceptable, since most protocols place the burden of proof on the party which made the claim.

In the rest of this paper we describe the work we have done so far to support argument evaluation, construction and visualization before concluding and discussing our plans for future work.

2 Argument Evaluation

We begin in the middle, dialectical layer of Figure 1, because it is central to our work, and not just in the diagram. Since the main task of the bottom, logical, layer, is to construct arguments, and the main task of the top, rhetorical, layer is to present arguments, we first need to define what we mean by arguments and how they are evaluated.

Informally, an argument links a set of statements, the premises, to another statement, the conclusion. The premises may be labelled with additional information, about their role in the argument. Aristotle's theory of syllogism, for

example, distinguished major premises from minor premises. The basic idea is that the premises provide some kind of support for the conclusion. If the premises are accepted, then the argument, if it is a good one, lends some weight to the conclusion. Unlike instances of valid inference rules of classical logic, the conclusion of an argument need not be necessarily true if the premises are true. Moreover, some of the premises of an argument may be implicit. An argument with implicit premises is called an *enthymeme* (Walton, 2006, p. 178).

We developed the mathematical model of argument which serves as the foundation for the Carneades software tools at the dialectical level in a series of papers (Gordon, 2005; Gordon and Walton, 2006; Gordon et al., 2007; Gordon and Walton, 2009a). Let us focus here on the later, more mature papers. In (Gordon et al., 2007) we presented a formal, mathematical model of argument structure and evaluation which applied proof standards to determine the acceptability of statements on an issue-by-issue basis. The model uses different types of premises (ordinary premises, assumptions and exceptions) and information about the dialectical status of statements (stated, questioned, accepted or rejected) to allow the burden of proof to be allocated to the proponent or the respondent, as appropriate, for each premise separately. Our approach allows the burden of proof for a premise to be assigned to a different party than the one who has the burden of proving the conclusion of the argument, and also to change the burden of proof or applicable proof standard as the dialogue progresses from stage to stage. Useful for modeling legal dialogues, the *burden of production* and *burden of persuasion* can be handled separately, with a different responsible party and applicable proof standard for each. Finally, following Verheij (2003), we showed another way to formally model critical questions of argumentation schemes as additional premises, using premise types to capture the varying effect on the burden of proof of different kinds of questions.

In Gordon and Walton (2009a), we developed this model further, with the aim of integrating the features of prior computational models of proof burdens and standards, in particular the model of Prakken and Sartor (2009) into Carneades. The notions of *proof standards* and *burden of proof* are relevant only when argumentation is viewed as a dialogical process for making justified decisions. During such dialogues, a theory of the domain and proofs showing how propositions are supported by the theory are collaboratively constructed. The concept of proof in this context is weaker than it is in mathematics. A proof is a structure which enables an audience to decide whether a proposition statisfies some proof standard, where a proof standard is a method for aggregating or accruing arguments. There are a range of proof standards, from *scintilla of evidence* to *beyond reasonable doubt* in the law, ordered by their strictness. The applicable standards depend on the issue and the type of dialogue, taking into consideration the risks of making an error. Whereas finding or constructing a proof can be a hard problem, checking the proof should be an easy (tractable) problem, since putting the proof into a comprehensible form is part of the burden and not the responsibility of the audience. Argumentation dialogues progress through three phases and different proof burdens apply at each phase: The burdens of claiming and questioning apply in the opening

phase; the burden of production and the tactical burden of proof apply in the argumentation phase; and the burden of persuasion applies in the closing phase.

The Carneades software, which is implemented in a functional style using the Scheme programming langauge (Dybvig, 2003), enables arguments and argument graphs to be represented and proof standards to be assigned to statements in a graph. Argument graphs are immutable and all operations on argument graphs are non-destructive, as dictated by the functional programming paradigm. Every modification to an argument graph, such as asserting or deleting an argument, or changing the proof standard assigned to a statement, returns a new argument graph, leaving the original unchanged. The acceptability of statements in a graph is computed and, if necessary, updated at the time the graph is modified. Dependency management techniques, known from reason maintenance systems (Doyle, 1979; De Kleer, 1988), are used to minimize the amount of computation needed to update the labels of statements in the graph, as changes are made. Quering an argument graph, to determine the acceptability of some statement in the graph, just performs a lookup of the pre-computed label of the statement, and can be performed in constant time. An XML syntax for encoding and interchanging Carneades arguments, inspired by Araucaria's Argument Markup Language (Reed and Rowe, 2004), has been developed, as part of the Legal Knowledge Interchange Format (ESTRELLA Project, 2008). The Carneades software is able to import and export argument graphs in this LKIF format.

3 Argument Construction

Argumentation schemes are useful for reconstructing, classifying and evaluating arguments, after they have been put forward in dialogues, to check whether a scheme has been applied correctly, identify missing premises and ask appropriate critical questions. Argumentation schemes are also useful for constructing new arguments to put forward, by using them as templates, forms or, more generally, procedures for generating arguments which instantiate the pattern of the scheme. We elaborated the role of argumentation schemes for generating arguments in a series of papers (Gordon, 2007b, 2008; Gordon and Walton, 2009b), focusing on computational models of argumentation schemes studied in the field of Artificial Intelligence and Law for legal reasoning, including Argument from Defeasible Rules, Argument from Ontologies, and Argument from Cases.

3.1 Argument from Rules

The term "rule" has different meanings in different fields, such as law and computer science. The common sense, dictionary meaning of rule (Abate and Jewell, 2001) is "One of a set of explicit or understood regulations or principles governing conduct within a particular sphere of activity." It is this kind of rule that we are interested in modeling for the purpose of constructing arguments. In the field of artificial intelligence and law, there is now much agreement about

the structure and properties of rules of this type (Gordon, 1995; Prakken and Sartor, 1996; Hage, 1997; Verheij, 1996):

1. Rules have properties, such as their date of enactment, jurisdiction and authority.
2. When the antecedent of a rule is satisfied by the facts of a case, the conclusion of the rule is only presumably true, not necessarily true.
3. Rules are subject to exceptions.
4. Rules can conflict.
5. Some rule conflicts can be resolved using rules about rule priorities, e.g. *lex superior*, which gives priority to the rule from the higher authority.
6. Exclusionary rules provide one way to undercut other rules.
7. Rules can be invalid or become invalid. Deleting invalid rules is not an option when it is necessary to reason retroactively with rules which were valid at various times over a course of events.
8. Rules do not counterpose. If some conclusion of a rule is not true, the rule does not sanction any inferences about the truth of its premises.

One consequence of these properites is that rules cannot be modeled adequately as material implications in predicate logic. Rules need to be *reified* as terms, not formulas, so as to allow their properties, e.g. date of enactment, to be expressed and reasoned about for determing their validity and priority.

In the Carneades software, methods from logic programming have been adapted and extended to model legal rules and build an inference engine which can construct arguments from rules. Rules in logic programming are Horn clauses, i.e. formulas of first-order logic in disjunctive normal form, consisting of exactly one positive literal and zero or more negative literals. The positive literal is called the 'head' of the rule. The negative literals make up the 'body' of the rule. A rule with an empty body is called a 'fact'. In logic programming these rules are interpreted as material conditionals in first-order logic and a single inference rule, resolution, is used to derive inferences. Since there is no way to represent negative facts using Horn clauses, rules do not counterpose in logic programming, even though they are interpreted as material conditionals and the resolution inference rule is strong enough to simulate modus tollens. In Carneades, we do not interpret rules as material conditionals, but as domain-dependent inference rules. Both the head and body of rules are more general than they are in Horn clauses. The head of a Carneades rule consists of a set of literals, i.e. both positive and negative literals. The body of a Carneades rule consists an arbitrary first-order logic formula, except that quantifiers and biconditionals are not supported. Variables in the body and head of a rule are interpreted as schema variables. Using de Morgan's laws, Carneades compiles rules into clauses in disjunctive normal form. Given an atomic proposition P, a

rule can be used to construct an argument pro or con P if P or $\neg P$, respectively, can be unified with a literal in the head of the rule.

The burden of proof for an atomic proposition in the body of a rule can be allocated to the opponent of the argument constructed using the rule, by declaring the proposition to be an exception. The syntax of rules has been extended to allow such declarations. Similarly, a proposition in the body of a rule can be made assumable, without proof, until it has been questioned by the opponent of the argument. These features make it possible to use Carnedes rules to model a broad range of argumentation schemes, where exceptions and assumptions are used to model the critical questions of the scheme. Whether a critical question should be modeled as an exception or an assumption depends on whether the "shifting burden" or the "backup evidence" theory of critical questions is more appropriate (Gordon et al., 2007).

The Carneades inference engine uses rules to construct and search a space of argument *states*, where each state consists of:

topic. The statement, i.e. proposition, which is the main issue of the dialogue, as claimed by its proponent.

viewpoint. Either 'pro' or 'con'. When the viewpoint is pro, the state is a goal state if and only if the topic of the state satisfies its proof standard. If the viewpoint is con, the state is a goal state only if the topic does not satisfy its proof standard. Notice the asymmetry between pro and con. The con viewpoint need not prove the complement of the topic, but need only prevent the pro viewpoint from achieving its goal of proving the topic.

pro-goals. A list of clauses, in disjunctive normal form, where each clause represents a set of statements which might be useful for helping the proponent to prove the topic.

con-goals. A list of clauses, in disjunctive normal form, where each clause represents a set of statements which might be useful for helping the opponent to prevent the proponent from proving the topic.

arguments. A graph of the arguments, representing all the arguments which have been put forward, hypothetically, by both the pro and con roles during the search for arguments.

substitution. A substitution environment mapping schema variables to terms. The scope of variables is the whole argument graph. Variables in rules are renamed to prevent name conflicts when they are applied to construct arguments.

candidates. A list of candidate arguments, which have been previously constructed. A candidate argument is added to the argument graph, and removed from this list, only after all of its schema variables are instantiated in the substitution environment. This assures that all statements in the argument graph are ground atomic formulas.

The space of states induced by a set of argument generators, such as the generator for the scheme for argument from rules, may be infinite. Carneades is implemented in a modular way which allows the space to be searched using any heuristic search strategy. Common strategies have been implemented, including depth-first search, breadth-first and iterative-deepening. For all of these strategies a resource limit, restricting the number of states which may be visited in the search space, may be specified, to assure termination of the search procedure. The system is extensible. Further heursitic-search strategies, including domain-dependent strategies, can be implemented and plugged into the search procedure. By default, Carneades uses a resource-limited version of depth-first search.

The heuristic search strategies are implemented in a purely functional way. Given a state and set of argument generators for computing successor states, the successor states are modeled as a stream of states, where each successor state is generated, lazily, just before it is visited by the search strategy to check whether or not it is a goal state.

The Legal Knowledge Interchange Format (LKIF) also includes an XML language for rules, as well as arguments (ESTRELLA Project, 2008). The Carneades software is able to import and export both arguments and rules in LKIF format.

3.2 Argument from Ontologies

In computer science, an *ontology* is a representation of concepts and relations among concepts, typically expressed in some decidable subset of first-order logic, such as description logic (Baader et al., 2003). Such ontologies play an important role in integrating systems, by providing a formal mechanism for sharing terminology, and also in the context of the so-called *Semantic Web* (Berners-Lee et al., 2001) for providing machine-processable meta-data about web resources and services. There is a World Wide Web standard for modeling ontologies in XML, called the Web Ontology Langauge (OWL) (McGuinness and van Harmelen, 2004). The Carneades software includes a compiler from OWL ontologies, encoded using the syntax of the Knowledge Representation System Specification (KRSS) (Patel-Schneider and Swartout, 1994), into Carneades rules, based on the Description Logic Programming (DLP) mapping of Description Logic axioms into Horn clause rules (Grosof et al., 2003). The latest version of OWL includes a rule language profile, called "OWL 2 RL", which is also based on DLP. We may in the future make an effort to assure that Carneades is compliant with the RL profile of the OWL 2 standard. Work is in progress on a compiler from the standard RDF/XML format for OWL directly into Carneades rules, to avoid the currently necessary intermediate step of translating OWL ontologies in RDF/XML first into KRSS format.

Ontologies and rules may be used together to construct arguments with Carneades. LKIF uses OWL to define the language of individual, predicate and function symbols, represented as Uniform Resource Identifers (URIs), which may be used in rules. URIs provide a world-wide way to manage symbols,

avoiding ambiguity and name clashes. This enables very large knowledge bases to be constructed, in a distributed and modular way. LKIF files can import OWL ontologies and, recursively, other LKIF files.

Reasoning with OWL ontologies is defeasible in Carneades. Any argument derived from axioms of an ontology, using the translation of these axioms into Carneades rules, may be undercut or rebut by other arguments. Thus, unlike some nonmonotonic logics, such as Defeasible Logic (Nute, 1994), Carneades does not distinguish between 'strict' and defeasible rules. All rules are defeasible, also those derived from the axioms of an ontology. We believe this simplification is justified for the kinds of domains we are interested in supporting, where practical reasoning about the "real world" is more important than reasoning about abstract mathematical concepts. Outside of mathematics, generalizations are rarely universal.

3.3 Argument from Cases

In the Artificial Intelligence and Law community a great deal of research has been conducted on case-based legal reasoning, particularly in common law jurisdiction such as the United States, where arguing with precedent cases plays a very central role in both legal education and practice. When trying to apply legislation to a particular case, at least two kinds of related interpretation problems must be faced. The first interpretation problem is to construct general legal rules from legislation expressed in natural language. The goal is to determine the operative legal facts which must be established when applying a legal rule in order to construct an argument for the conclusion of the rule. In the law of contracts, for example, there is a rule stating, essentially, that a contract requires an offer, acceptance and an exchange of promises ('consideration'). Here, 'offer', 'acceptance' and 'consideration' are the operative legal facts. They are abstract technical terms of law. The second interpretation problem arises when trying to apply these operational legal facts to the concrete facts of a particular case. Has an offer been made? Has consideration been exchanged? This problem, called the subsumption problem, also requires the interpretation of the legislation, in the light of past precedent cases. Operative legal facts typically denote 'open-textured' concepts (Hart, 1961).

These interpretation problems are at the root of the continual debate about the proper role of judges and the relationship between the legislative and judicial branches of government, as we witnessed again recently during the confirmation hearings of Judge Sotomayor to the US Supreme Court. Since judges are not typically elected by the people, and may be thought to lack the same level of democratic legitimacy, many people insist that judges should interpret legislation narrowly. Judges should merely apply the law and not usurp the exclusive right of the legislature to make law. But an overly simplistic view of judging fails to appreciate the distinctions between legal norms and their necessarily imperfect and ambigious representation in natural language in legislation and the hermeneutic difficulties of interpreting natural language, even given the best of intentions and conscientious effort.

In the field of Artifical Intelligence and Law, computational models of case-based reasoning are arguably still at an early developmental stage, reflecting presumably the lack of a deep and widely accepted theory of legal reasoning with cases in legal philosophy. The most influential work on case-based reasoning in AI and Law is the HYPO system (Rissland, 1989; Ashley, 1990) and its successors, CABARET (Skalak and Rissland, 1992) and CATO (Aleven and Ashley, 1997).

In Carneades, we have implemented a reconstruction of CATO by Wyner and Bench-Capon (2007). For some legal issue, such as whether or not information was a trade secret, the legal domain modeled in HYPO, the precedent cases are analysed to collect the set of *factors* which were found relevant for deciding the issue. A factor is a proposition which tends to favor either one side or the other regarding the legal issue. For example, efforts to keep the information secret tend to support a finding that the information was a trade secret, whereas disclosure of the information to people outside the company tends to support a finding that the information was not a trade secret. Each precedent case is modeled as a set of such factors together with the decision of the court regarding the issue of interest, such as, in our example, whether the information was found to be a trade secret or not. Given the set of factors known or assumed to be true in the current case, Carneades constructs, for each precendent case, a set of six partitions of the union of the factors of the current case and the precedent case. For example, partition P1 is the intersetion of the pro-plaintiff factors in the precedent case and the pro-plaintiff factors in the current case. And partition P5 is the set of pro-defendant factors in the current case which are not in the precedent case. These partitions are used in computational models of six argumentation schemes to analogize cases, distinguish cases argued to be analogous as well as downplay these distinctions.

There is a problem integrating arguments from ontologies or rules with case-based arguments using this reconstruction of CATO, since ontologies and rules are modeled at a finer level of granularity, using predicate logic, than the factors in this model of case-based reasoning, which are at a more abstract, propositional level of granularity. This problem is overcome in Carneades using bridging rules, similar to the output/input transformers of Prakken (2008), to map predicate logic formulas to factors. Since different instantiations of schema variables in the predicate logic formulas can get mapped to the same factor, losing any distinctions between these various formulas in the resulting argument graphs, this solution only works when one can safely assume that at most one predicate logic formula will get mapped to each factor during the analysis of a particular case. For example, this solution may not be adequate if two different pieces of information must be evaluated to be determined whether they are trade secrets in a single case, unless the problem can be reduced to two problems that can be analysed separately.

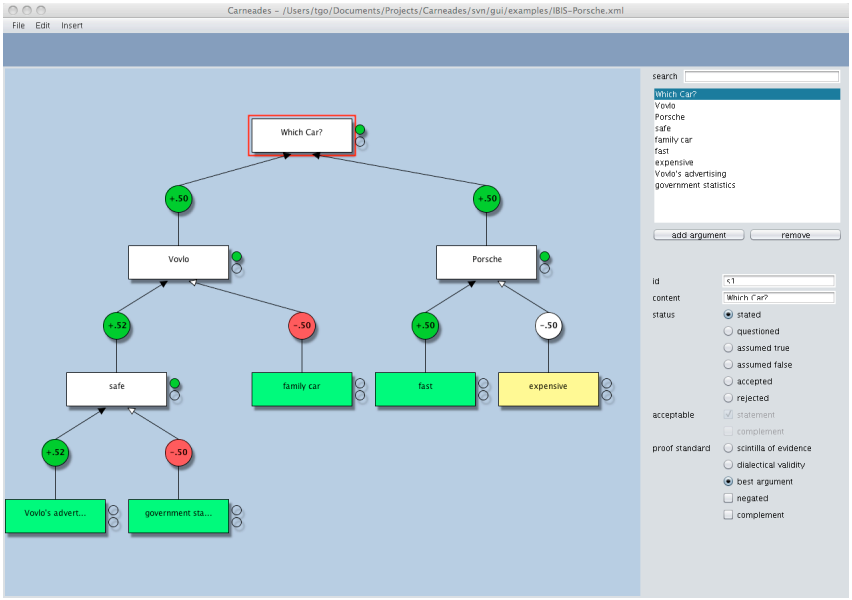

Figure 2: Screenshot of the Carneades Argument Diagramming Tool

4 Argument Visualization

We have been experimenting with methods for visualizing Carneades argument graphs and designing graphical user interfaces for working with argument graphs. An important difference between our work and most prior work on argument visualization, with the exception of Verheij (2005), is that our diagrams are views onto a mathematical model of argument graphs and the user interfaces provide ways to modify, control and view the underlying model. Argument diagramming software for Wigmore (1908), Beardsley (1950) or Toulmin (1958) diagrams, such as Araucaria (Reed and Rowe, 2004) or Rationale[3], lack this mathematical foundation. Essentially, the diagrams are the models in these other systems, rather than views onto a model.

Our approach gives us much freedom to experiment with different diagramming methods and user interfaces for manipulating Carneades argument graphs, without changing the underlying model of argument. In Gordon (2007a), we described a couple of different approaches, including one which is very close to Wigmore's style of argument diagramming.

One diagramming issue we have been discussing again recently, is how best to represent undercutters, i.e arguments which directly attack the inferential link between the premises of an argument and its conclusion. An undercutter claims that the premises of the argument being undercut do not give us reason to derive the conclusion of the argument, not even presumptively. In the

[3]http://rationale.austhink.com/

Carneades mathematical model of argument graphs, undercutters are modeled as attacks on the major premise of an argument. If the major premise has been left implicit, in an enthymeme, then it is first made explicit by adding it to the argument, before adding the undercutting argument. Some diagramming tools, such as ArguMed (Verheij, 2005), use a technique called 'entanglement' to visualize undercutters, in which the undercutting argument points to the arrow between the premises and conclusion of the argument being undercut. It would be possible to use this diagramming method in views of Carneades argument graphs as well, by visualizing the major premise of an argument as the arrow or link from the minor premises to the conclusion of the argument.

The Carneades software includes a library for generating diagrams of argument graphs using Graphviz (Ellson et al., 2001), which can produce graphs in various file formats, including PDF and SVG[4]. These diagrams, however, are not interactive and thus do not provide a user interface for creating or editing argument graphs. Work is underway on an new argument diagramming tool for Carneades which will be much more interactive, provide better support for argumentation schemes and be integrated with other Carneades tools, such as the inference engine for generating arguments from ontologies, rules and cases. Figure 2 shows a screen shot of a prototype of this new tool. The Carneades argument diagramming tool will provide users with a variety of views onto Carneades argument graphs, including high-level, abstract views which show only attack and support relations among arguments, similar to diagrams of the kind typically used to visualize Dung Argumentation Frameworks (Dung, 1995).

5 Conclusions and Future Work

The Carneades software is a toolbox for supporting various argumentation tasks, designed and built in collaboration with Doug Walton on the foundation of his philosophy of argumentation. The Carneades argumentation toolbox is work in progress. Here we summarized the tools currently available in Carneades for reconstructing, representing and evaluating arguments, for using a knowledge-base of ontologies, rules and case to search for and construct arguments for both sides of an issue and for visualizing complex networks of arguments in any effort to make relationships among arguments clearer and more understandable, especially to lay audiences.

Much work remains. Although the Carneades inference engine for the LKIF was validated in pilot applications during the ESTRELLA project, further pilot applications and case studies are needed, especially with regard to other parts of the system, such as the argument visualization tools. Currently the system is too difficult to install and requires too much specialist knowledge to use. Our work on the new argument diagramming tool is a first step in the direction of developing a better integrated, easy-to-install and easy-to-use version of Carneades.

[4] http://www.w3.org/Graphics/SVG/

In addition to consolidating existing features, improving their usability with a rich graphical user-interface, we have plans for additional features, to support further argumentation tasks:

- The next version of the Carneades argument diagramming tool will include support for using argumentation schemes, both to classify and evaluate existing arguments and to construct new arguments, similar to the support for argumentation schemes provided currently by Araucaria.

- As discussed in (Gordon and Walton, 2009b), Carneades currently also has a module for interacting with users to generate arguments from witness testimony. This serves as a dialogue component for Carneades which enables it to be used as an expert-system shell. As Carneades applies ontologies, rules and cases while searching for arguments to answer a query posed by the user, in a backward-chaining way, this component enables Carneades to ask the user questions to obtain information about the facts of the case. This component currently only has a command-line user interface and is thus little more than a proof-of-concept. A graphical user-interface for this feature is planned, for use in both web and desktop applications.

- Similarly, we would like to extend Carneades with a web-based user interface comparable to Parmenides (Atkinson et al., 2006), to enable Carneades to be used as a collaboration tool on the web for collecting arguments, by generating surveys which help lay users to apply argumentation schemes and ask critical questions.

- The argument diagramming tool and the inference engine tool for generating arguments from knowledge-bases should be more tightly integrated, to implement a kind of graphical debugger and tracer. As the inference engine searches for arguments, the argument graph of each state in the search space could be visualized, along with the other information of the state, such as the list of open goals. We would like to provide users with a way to guide the search for arguments, for example by manually ordering the goals or backtracking to some prior state. This system would take us a step further towards our goal of making Carneades a tool for assisting users with argumentation tasks. Currently the argument construction module is too much like a fully automatic theorem prover.

- We would like to continue our collaboration with Lauritsen on document assembly to extend Carneades with a tool which uses the information in an argument graph to generate outlines of explanatory or justificatory documents, such as court opinions (Lauritsen and Gordon, 2009).

- At the rhetorical level, we have begun work on modeling assumptions about the beliefs, values and preferences of audiences, along with methods, using a kind of abduction, which make use of this information to select goal statements to try to prove or disprove in dialogues.

- One of the anonymous reviewers suggested an interesting topic for future research: modeling knowledge about whether it is worthwhile to engage in argumentation in the first place. As he pointed out, arguing is not always the best way to resolve conflicts or coordinate actions.
- We are working with Brewka to apply his work on Abstract Dialectical Frameworks (Brewk and Woltran, 2010) to clarify the formal relationship between Carneades and Dung Argumentation Frameworks (Dung, 1995).
- Finally, we plan to revisit the research topic we addressed in the Pleadings Game (Gordon, 1995), by extending Carneades with tools supporting the procedural aspects of argumentation dialogues. But unlike the Pleadings Game, which modeling a particular type of dialogue, our aim here is to provide Carneades with a way to define and use protocols for a variety of dialogue types. We would like to investigate the suitability of prior work on business process modeling tools and workflow engines for this purpose.

6 Acknowledgements

The work reported here was supported in part by a grant from the European Commission, in the ESTRELLA project (IST-4-027655). I would like to acknowledge my colleague, Stefan Ballnat, for his substantial contributions to the implementation of the Careades inference engine and Matthias Grabmair for his work on the implementation of the new argument diagramming tool for Carneades. Finally, I would like to thank Doug Walton. It has been an honor and a pleasure to be able to collaborate with him closely the last few years.

References

Frank Abate and Elizabeth J. Jewell, editors. *New Oxford American Dictionary.* Oxford University Press, 2001.

Vincent Aleven and Kevin D. Ashley. Evaluating a learning environment for case-based argumentation skills. In *ICAIL '97: Proceedings of the 6th international conference on Artificial intelligence and law*, pages 170–179, New York, 1997. ACM Press.

Kevin D. Ashley. *Modeling Legal Argument: Reasoning with Cases and Hypotheticals.* Artificial Intelligence and Legal Reasoning Series. MIT Press, Bradford Books, 1990.

K. Atkinson, T. Bench-Capon, and P. McBurney. PARMENIDES: facilitating deliberation in democracies. *Artificial Intelligence and Law*, 14(4):261–275, 2006.

Franz Baader, Diego Calvanese, Deborah McGuinness, Daniele Nardi, and Peter Patel-Schneider, editors. *The Description Logic Handbook – Theory, Implementation and Applications*. Cambridge University Press, 2003.

Monroe C. Beardsley. *Practical Logic*. Prentice Hall, New York, 1950.

Tim Berners-Lee, James Hendler, and Ora Lassila. The semantic web. *Scientific American*, 284(5):34–43, May 2001.

Gerhard Brewk and Stefan Woltran. Abstract dialectical frameworks. In *Proceedings of the Twelfth International Conference on the Principles of Knowledge Representation and Reasoning*. in press, AAAI Press, 2010.

Gerhard Brewka and Thomas 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.

J. De Kleer. A general labeling algorithm for assumption-based truth maintenance. In *Proceedings of the 7th national conference on artificial intelligence*, pages 188–192, San Francisco, California, USA, 1988. Morgan Kaufmanns Publishers.

Jon Doyle. A truth maintenance system. *Artificial Intelligence*, 12:231–272, 1979.

Phan Minh 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. ISSN 0004-3702.

R. Kent Dybvig. *The Scheme Programming Language*. MIT Press, third edition edition, 2003.

John Ellson, Emden Gansner, Lefteris Koutsofios, Stepen C. North, and Gordon Woodhull. Graphviz — open source graph drawing tools. In *Proceedings of the 9th International Symposium on Graph Drawing (GD 2001)*, pages 483–484, Vienna, September 2001.

ESTRELLA Project. The legal knowledge interchange format (LKIF). Deliverable 4.1, European Commission, 2008.

Thomas F. Gordon. *The Pleadings Game; An Artificial Intelligence Model of Procedural Justice*. Springer, New York, 1995. Book version of 1993 Ph.D. Thesis; University of Darmstadt.

Thomas F. Gordon. A computational model of argument for legal reasoning support systems. In Paul E. Dunne and Trevor Bench-Capon, editors, *Argumentation in Artificial Intelligence and Law*, IAAIL Workshop Series, pages 53–64, Nijmegen, The Netherlands, 2005. Wolf Legal Publishers.

Thomas F. Gordon. Visualizing Carneades argument graphs. *Law, Probability and Risk*, 6(1-4):109–117, 2007a.

Thomas F. Gordon. Constructing arguments with a computational model of an argumentation scheme for legal rules. In *Proceedings of the Eleventh International Conference on Artificial Intelligence and Law*, pages 117–121, 2007b.

Thomas F. Gordon. Hybrid reasoning with argumentation schemes. In *Proceedings of the 8th Workshop on Computational Models of Natural Argument (CMNA 08)*, pages 16–25, Patras, Greece, July 2008. The 18th European Conference on Artificial Intelligence (ECAI 2008).

Thomas F. Gordon and Douglas Walton. The Carneades argumentation framework — using presumptions and exceptions to model critical questions. In Paul E. Dunne and Trevor J.M. Bench-Capon, editors, *Computational Models of Argument. Proceedings of COMMA 2006*, pages 195–207, Amsterdam, September 2006. IOS Press.

Thomas F. Gordon and Douglas Walton. Proof burdens and standards. In Iyad Rahwan and Guillermo Simari, editors, *Argumentation in Artificial Intelligence*. Springer-Verlag, Berlin, Germany, 2009a.

Thomas F. Gordon and Douglas Walton. Legal reasoning with argumentation schemes. In Carole D. Hafner, editor, *12th International Conference on Artificial Intelligence and Law (ICAIL 2009)*, New York, NY, USA, 2009b. ACM Press.

Thomas F. Gordon, Henry Prakken, and Douglas Walton. The Carneades model of argument and burden of proof. *Artificial Intelligence*, 171(10-11):875–896, 2007.

Benjamin N. Grosof, Ian Horrocks, Raphael Volz, and Stefan Decker. Description logic programs: Combining logic programs with description logics. In *Proceedings of the Twelth International World Wide Web Conference (WWW 2003)*, pages 48–57, Budapest, Hungary, May 2003. ACM.

Jaap C. Hage. *Reasoning with Rules – An Essay on Legal Reasoning and its Underlying Logic*. Kluwer Academic Publishers, Dordrecht, 1997.

H. L. A. Hart. *The Concept of Law*. Clarendon Press, Oxford, 1961.

Marc Lauritsen and Thomas F. Gordon. Toward a general theory of document modeling. In Carole D. Hafner, editor, *12th International Conference on Artificial Intelligence and Law (ICAIL 2009)*, New York, NY, USA, 2009. ACM Press.

Ann Macintosh, Thomas F. Gordon, and Alastair Renton. Providing argument support for e-participation. *Journal of Information Technology & Politics*, 6(1):43–59, 2009. URL `http://www.informaworld.com/10.1080/19331680802662113`.

Deborah L. McGuinness and Frank van Harmelen. OWL Web Ontology Language overview. http://www.w3.org/TR/owl-features/, 2004.

Donald Nute. Defeasible logic. In D. Gabbay, C.J. Hogger, and J.A. Robinson, editors, *Handbook of Logic in Artificial Intelligence and Logic Programming*, pages 253–395. Clarendon Press, Oxford, 1994.

P.F. Patel-Schneider and B. Swartout. Description logic knowledge representation system specification from the KRSS group of the ARPA knowledge sharing effort. Report, AT&T Bell Laboratories, Murray Hill, New Jersey, 1994.

Henry 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.

Henry Prakken. Combining modes of reasoning: An application of abstract argumentation. In *Proceedings of The 11th European Conference on Logics in Artificial Intelligence (JELIA 2008)*, volume 5293 of *Springer Lecture Notes in AI*, pages 349–361, Berlin, April 2008. Springer Verlag.

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

Henry Prakken and Giovanni Sartor. A logical analysis of burdens of proof. In H. Kaptein, Henry Prakken, and Bart Verheij, editors, *Legal Evidence and Proof: Statistics, Stories, Logic*, Applied Legal Philosophy Series, pages 223–253. Ashgate Publishing, 2009.

Chris A. Reed and Glenn W.A. Rowe. Araucaria: Software for argument analysis, diagramming and representation. *International Journal of AI Tools*, 13 (4):961–980, 2004.

Edwina L. Rissland. Dimension-based analysis of hypotheticals from supreme court oral argument. In *The Second International Conference on Artificial Intelligence and Law*, pages 111–120. ACM Press, 1989.

David B. Skalak and Edwina L. Rissland. Arguments and cases: An inevitable intertwining. *Aritificial Intelligence and Law*, 1(1):3–45, 1992.

Stephen E. Toulmin. *The Uses of Argument*. Cambridge University Press, Cambridge, UK, 1958.

Bart Verheij. *Rules, Reasons, Arguments. Formal Studies of Argumentation and Defeat*. Ph.d., Universiteit Maastricht, 1996.

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

Bart Verheij. *Virtual Arguments*. TMC Asser Press, The Hague, 2005.

Douglas Walton. *The New Dialectic: Conversational Contexts of Argument*. University of Toronto Press, Toronto; Buffalo, 1998. 24 cm.

Douglas Walton. *Fundamentals of Critical Argumentation*. Cambridge University Press, Cambridge, UK, 2006.

Douglas Walton, Chris Reed, and Fabrizio Macagno. *Argumentation Schemes*. Cambridge University Press, 2008.

John H. Wigmore. *A Treatise on the System of Evidence in Trials at Common Law: Including the Statutes and Judicial Decisions of all Jurisdictions of the United States*. Little, Brown and Company, Boston, Massachusetts, USA, 1908.

Adam Wyner and Trevor Bench-Capon. Argument schemes for legal case-based reasoning. In *JURIX 2007: The Twentieth Annual Conference on Legal Knowledge and Information Systems*, 2007.