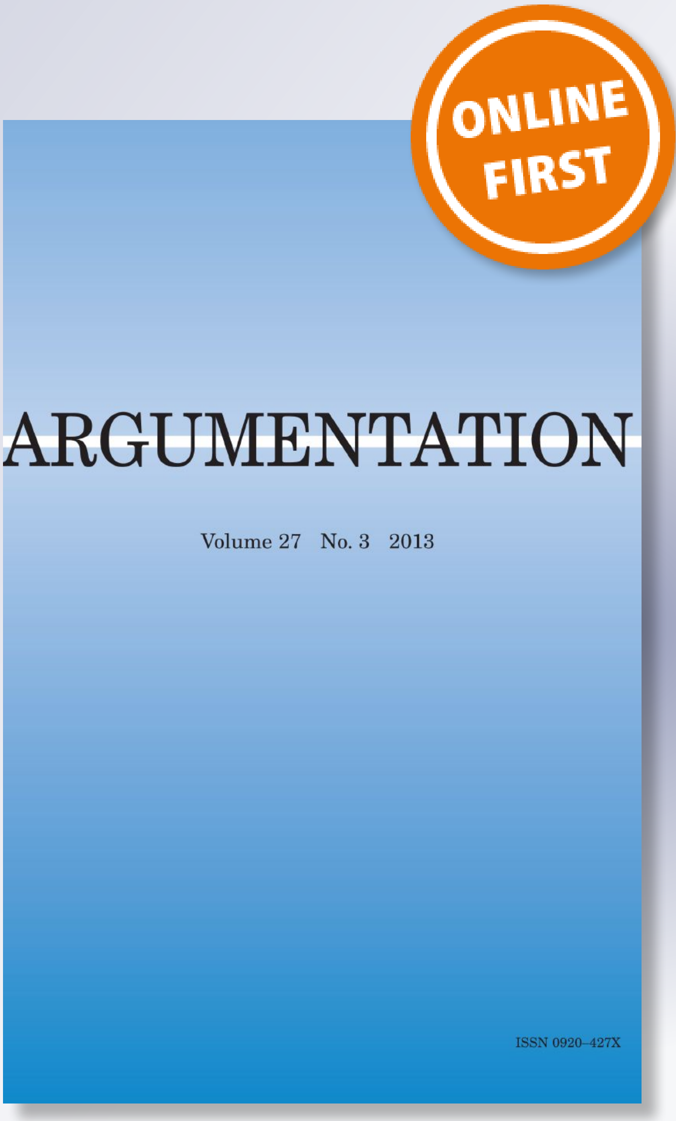
How Computational Tools Can Help Rhetoric and Informal Logic with Argument Invention

Douglas Walton & Thomas F. Gordon

Argumentation
An International Journal on Reasoning

ISSN 0920-427X

Argumentation
DOI 10.1007/s10503-017-9439-5



Springer

Argumentation
https://doi.org/10.1007/s10503-017-9439-5

How Computational Tools Can Help Rhetoric and Informal Logic with Argument Invention

Douglas Walton[1] · Thomas F. Gordon[2]

Abstract This paper compares the features and methods of the two leading implemented systems that offer a tool for helping a user to find or invent arguments to support or attack a designated conclusion, the Carneades Argumentation System and the IBM Watson Debater tool. The central aim is to contribute to the understanding of scholars in informal logic, rhetoric and argumentation on how these two software systems can be useful for them. One contribution of the paper is to explain to these potential users how the two tools are applicable to the task of inventing arguments by using some simple illustrative examples. Another is to redefine the structure of argument invention as a procedure.

Keywords Rhetorical invention · Artificial intelligence · Argument mining · Computational linguistics

1 Introduction

It is well-known that the ancient rhetoricians devised methods to help an arguer skillfully take part in public debates by applying systematic methods to enable the finding of new arguments to support or attack a claim (Cicero 1949; Kennedy 1963). Attempts to refine these rhetorical methods and build on them to try to develop

✉ Douglas Walton
dwalton@uwindsor.ca
http://www.dougwalton.ca

Thomas F. Gordon
thomas.gordon@fokus.fraunhofer.de

1 Centre for Research on Reasoning, Argumentation and Rhetoric, University of Windsor, 401 Sunset Avenue, Windsor, ON N9B 3P4, Canada

2 Fraunhofer FOKUS, Kaiserin-Augusta-Allee 31, 10589 Berlin, Germany

Published online: 27 January 2018

systems of argument invention continued through medieval times, but interest in them lapsed as rhetorical invention began to be seen during the Enlightenment period as a purely humanistic skill not amenable to scientific calculation. The popular logic textbook called the *Port-Royal Logic* (Arnauld 1662) spread what came to be the dominant view that such methodical systems to assist argument invention are neither necessary nor useful (Kienpointner 1997, 228). On Arnauld's "encyclopedic" view of argument invention, all that is needed is a good memory of the facts (Kienpointner 1997, 230). Nowadays however, we have found that human memory is limited, and that a computerized database is a much more accurate and reliable way of storing big data. We have also learned how to devise search engines to scan through large databases to find and present facts on a topic with amazing speed and accuracy. We are in an era in which artificial intelligence is able to help arguers by providing new automated argument assistants that can speedily and reliably assist a persuasion attempt.

Aristotle defined rhetoric in the *Rhetoric* (Aristotle 1991, I.26–35) as the faculty of observing the available means of persuasion in any given case. He took rhetoric to be the power of observing the means of persuasion on almost any subject presented to us. The means is to find or invent arguments that can be used to persuade the audience. This paper compares the features, methods and underlying computational structures of the two leading implemented computational systems that offer devices for helping a user to find or invent arguments supporting a designated conclusion, the Carneades Argumentation System and the IBM Watson Debater tool. Both purport to be tools for finding arguments that can be used in a persuasion attempt. But it needs to be noted that the rhetorician has many other tools at his/her disposal. So, as will be shown, these new tools should not be seen as replacing the rhetorical skills of the trained and experienced speaker.

Before proceeding further however, 'finding' needs to be carefully distinguished from 'inventing'. To find something is to retrieve something that was already there somewhere, whereas to invent something is to come up with something new, something that was not there before. Finding arguments in a natural language text is the task of a branch of argumentation studies called argument mining. Argument mining is a new initiative in corpus-based discourse analysis that has the objective of identifying arguments within a natural language document. After explaining how the Debater tool works (Sect. 2), we will compare it to argument mining in Sect. 3. In Sect. 2 it is shown how Debater builds on techniques developed in natural language processing. In Sect. 4 it is shown how Carneades can perform the task of automated argument invention, by using its scheme/rule engine to build new arguments. Section 5 shows how argumentation schemes are applied in Carneades. Section 5 raises questions about the applicability of argumentation schemes to the task of argument invention, and gives an example to show that there are difficulties with having users begin with reconstructing arguments from real texts using argument diagrams. Section 6 offers a short example to explain the procedure that Carneades uses to invent arguments. Section 7 draws out ten specific points of comparison between Debater and Carneades, and Sect. 8 explains underlying structural differences between the two systems at a deeper level. Section 9 summarizes the strengths and weaknesses of each of the two systems and shows how they fit together, leading to some suggestions on directions new work on argument invention might take. Section 10 draws some conclusions and briefly discusses challenges for building systems of argument invention that combine the virtues of both systems.

2 The Watson Debater Tool

Levy et al. (2014) reported results of experiments in unsupervised learning designed to begin building a computational tool to assist a human user to find a set of convincing arguments pro or con in relation to the topic set for discussion. The topic is defined as a short phrase that frames the discussion (Levy et al. 2014, 1489). The tool they have devised helps a human arguer to search for statements that can be used to directly support or attack the given topic. The target of such a search is called a context dependent claim (CDC), defined as a general, concise statement that directly supports or contests a topic. The tool was developed and tested by having a team of human labelers search for CDC's in a selected collection of Wikipedia articles.

In a video demonstration of Debater,[1] the proposition 'The sale of violent videogames to minors should be banned' was put into the system as a topic. After scanning 4 million Wikipedia articles, Debater found 3000 sentences in the ten most relevant articles, collected the strongest arguments, and assessed the polarity of each of them as pro or con the topic using a technique called sentiment analysis (Turney 2002). Sentiment analysis is a technique of natural language processing developed in computational linguistics to extract subjective information from a text. It is used to determine the attitude, pro or con (polarity) of a sentence found in a text. Sentiment analysis is also used to determine the polarity of a larger body of text, such as a document. Categories other than pro and con can also be used, such as neutral, angry, or happy. Research in this area has studied examples of detecting polarity in product reviews and movie reviews. The four main techniques used in sentiment analysis are keyword spotting, lexical affinity, statistical methods and concept-level techniques. Keyword spotting identifies affect words such as 'superb' or 'nasty'. Lexical affinity assigns a probable affinity of a word to a given emotion. Statistical methods use such devices as detecting relationships between the author of the text and the entity which is the subject of the sentiment. Concept-level techniques detect subtle relationships between concepts using devices such as ontologies from knowledge representation. Sentiment analysis uses human coders along with these computational tools for extracting data from a text.

Figure 1 illustrates how the output works by showing a condensed version in the format of an argument diagram. As shown in Fig. 1, the output in this example is a set of three pro arguments and a set of three con arguments. The user gets this output immediately upon entering a topic and can choose which arguments to go with.

Debater went through a lengthy process of debugging as it was tested. One practical problem for the user trying to find arguments with Debater was that Wikipedia produced a large number of candidates CDC's initially, but only a small

[1] https://www.youtube.com/watch?v=sEf0GLvrP9U.

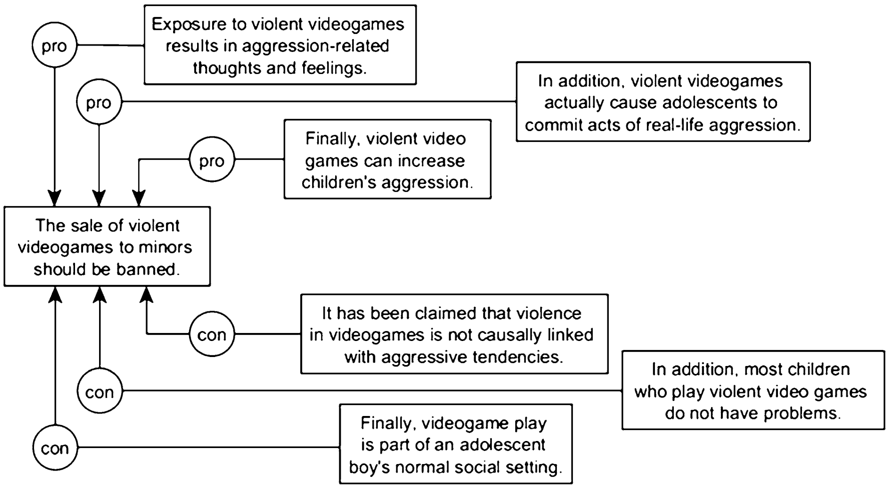


Fig. 1 Summary of the leading arguments found by Debater on the videogames topic

number of them were useful to an arguer who wants to find usable arguments on the pro or con side of the topic. The non-useful ones have to be selected out. To cite one example, the statement that violent video games can increase children's aggression was selected as a CDC, but the statement that violent video games should not be sold to children was also found as a CDC but had to be excluded. The reason why it was excluded is that it merely restates the topic, and so would not be useful to the arguer as a CDC.

A logical reason for excluding such a statement is that it merely produces a circular argument that would do nothing to support the claim made in the topic. Cases where the premise of an argument is closely equivalent to the conclusion, even though it uses different wording, are considered instances of the fallacy of begging the question in logic (Walton 1994). The reason is that such a premise can never increase the probative value of the conclusion. Only an argument where the premises have greater probative value than the conclusion is useful to increase the probative value of the conclusion.

To help solve these kinds of problems the Debater team worked with a group of in-house labelers who were trained rigorously to identify CDC's and exclude kinds of sentences not fitting the criteria for a CDC. They applied criteria described as strength, generality, phrasing, and topic unity (Levy et al. 2014, 1491). Strength indicates how strongly a sentence supports or contests the topic. Generality refers to content that deals with a relatively broad idea. Phrasing excludes sentences judged not to be grammatically correct or semantically coherent. Topic unity requires that the sentence should deal only with one topic, or not more than two topics.

Another illustration of the practical problem of using Debater to identify CDC's concerns the kind of case where a CDC is embedded into a longer Wikipedia sentence. In one of the examples they consider (Levy et al. 2014, 1489), the CDC claiming that violence in games hardens children to unethical acts is embedded in a longer sentence which states that two named individuals argue that violence in games hardens children to unethical acts, and goes on to call first-person shooter games murder simulators. This text contains several arguments combined in one long sentence where the CDC appears in the middle of the sentence. The task here is finding the boundaries of the text that can be classified as a CDC, so that the non-useful parts of a longer sentence in the natural language text can be excluded.

The high-level architecture of the Debater system uses seven engines that are currently at different stages of development (Aharoni et al. 2014, 8). The *topic analysis engine* is used to identify the main concepts mentioned in a topic and the sentiment towards each of these concepts. The *article retrieval engine* searches for Wikipedia articles that have a high probability of containing CDC's. The *CDC detection engine* zooms in within the retrieved articles to detect CDC's. The *CDC pro/con engine* automatically judges the polarity of a CDC found, with respect to a given topic. The *CDC equivalence engine* attempts to avoid redundancy by determining whether to CDC's are semantically equivalent. The *CDC refinement engine* has the task of improving the precision of the generated output, for example by removing judgments of pro-con polarity that have low confidence. Finally, the *text to speech engine* presents the top CDC candidates recommended as arguments for the user to consider pro or con the given topic.

Levy et al. (2014, 1491) see their ultimate goal as that of automatically pinpointing the CDC's within a set of natural language documents quickly so that the CDC that is found can be used right away. Timeliness in responding to an argument is often important. They adopted the approach that this goal can best be achieved by carrying out a prior task called context dependent claim detection (CDCD). Because they do not restrict this task to structured data or to a particular domain, three factors are important. One, as noted above, is to determine the boundaries of the CDC in a longer text. The second one is to determine the relevance of the CDC to the topic of the discussion. The third one is to determine, the persuasive impact of relevant argumentation, which is often based on an emotional impact that might or might not be relevant.

Debater is designed to find relevant arguments on a topic from a corpus that is a natural language text, namely Wikipedia in this instance, so a large part of the effort is directed to trying to find the CDC's by searching through the text, picking them out, and excluding the parts of the text that are not useful, even though they have been selected by the automated search system. Levy et al. (2014, 1491) describe this part of the process as the labeling task. To carry out this task they used in-house labelers, human agents that understand the natural language discourse and can sift through it to "funnel down" the data to useful statements that directly support or contest the topic of the discussion.

3 Argument Mining

Although argument mining and argument invention have different purposes, it would appear that the Watson Debater tool used for argument invention is very similar to the procedure used for argument mining. The purpose of argument invention, as stated in the previous section, is to help a user find arguments to support or attack a designated claim, a particular proposition taken as input to the system. The purpose of argument mining, as stated in the previous section, is to begin with a natural language text presumed to contain arguments, and search through the text to identify the arguments in it. Debater is supposed to be a tool for argument invention, but in fact what it does is to begin with a natural language text presumed to contain arguments, Wikipedia, designates a particular proposition as a topic, and searches through the text to identify arguments in it that support or attack the topic. Hence it would seem that the task that Debater performs is very similar to the kind of task to be carried out by the system of argument mining. Some might even think that Debater really is a system of argument mining, because it is so heavily based on using tools from computational linguistics to identify arguments in a natural language text. Argument mining needs to use the same tools, and also needs to rely on human argument assistants to debug problems in an effort to get more accurate results.

Although the methods of argument mining are in many instances the same as the methods used in argument invention, the purpose of each of these two research initiatives is quite different. The purpose of argument invention is to help a user find arguments to support or attack a designated claim, a particular proposition taken as input to the system. The purpose of argument mining is to begin with a natural language text presumed to contain arguments, and search through the text to identify arguments in it (Peldszus and Stede 2013). For example it would be extremely useful to be able to search through text of argumentative articles found in news sources, or in a document describing a Supreme Court case, and identify all the arguments from expert opinion or all the arguments from precedent in the text (Mochales and Moens 2011).

The goal of argument mining is to build a technology that can identify the premises, conclusion, and the argumentation scheme of each argument that is found in a text. Its capabilities are supposed to include the finding of subarguments under a given argument, pro arguments supporting a given argument, and also con arguments attacking a given argument (Anon 2014). There are many potential applications of this technology. It would enable information retrieval and could be linked to argument visualization tools to summarize arguments and provide argument diagrams for arguments found in a text. Most importantly, it could enable searching on the Internet, or through any large text of discourse, to find arguments of a particular type. Accomplishing this task would be very valuable for argumentation research.

It could also provide resources for assessing students' command of course material, and for using students' educational data to mine it for purposes of assessment and instruction.

Developing the sophisticated tools required for argumentation mining will require an interdisciplinary approach including natural language processing, pragmatics and discourse analysis, artificial intelligence, argumentation theory, and computational models of argumentation (Anon 2015). Discourse theories assume that the structure of a text is that of a graph or a tree, and that the elementary units of complex text structures are spans of text. The current systems being developed for use in argumentation mining look for prominent indicators of rhetorical structure expressed by conjunctions and certain kinds of adverbial groups. They identify words, as well as clusters of words, including pairs of successive words, adverbs, verbs and modal auxiliary verbs. Rhetorical structure theory defines rhetorical relations that can hold between spans of a text (Taboada and Mann 2006). Most hold between two text spans called a nucleus, the unit most central to the writer's purpose, and a satellite, which stands in a relation to the nucleus. To cite a legal example, the evidence relation links the nucleus 'Bob committed the murder' to the satellite 'Bob's DNA was found on the murder weapon'.

Existing work on argumentation mining has already made good use of natural language processing techniques that use human assistants to apply criteria to a natural language corpus to identify arguments in a text. Mochales and Moens (2011) have developed methods for classifying arguments in legal texts by collecting documents containing judges' legal decisions in a database of cases from the European Court of Human Rights. Argumentation schemes were used to help identify arguments. Phrases such as 'it follows that', and 'in conclusion', were used to identify conclusions. Phrases such as 'in the view of the factfinder' were used to identify premises of arguments. The task was helped by the fact that the documents they used were divided up into sections where one section was designated as containing summaries of the judges' arguments used to support their conclusions.

These results showed that applying argumentation schemes is a useful tool for identifying and analyzing kinds of arguments that are central to legal argumentation. For example, the scheme for argument from an established rule as configured by (Walton et al. 2008, 343) is shown below.

Major Premise: If carrying out types of actions including the state of affairs A is the established rule for x, then (unless the case is an exception), x must carry out A.
Minor Premise: Carrying out types of actions including state of affairs A is the established rule for a.
Conclusion: Therefore a must carry out A.

In the common kind of case in legal reasoning where an established rule is applied to a particular case by a judge this scheme can be expressed in a simpler form.

Major Premise: If rule R applies to facts F in case C, conclusion A follows.
Minor Premise: Rule R applies to facts F in case C.
Conclusion: In case C, conclusion A follows.

This scheme is comparable to the scheme for argument from a legal rule identified by Verheij (2008, 23).

Argument from precedent is another important scheme for of legal reasoning in the common law. In this type of argument, there is a case at issue, and a prior case that has already been decided is taken as a precedent that can be applied to the present case. The argumentation scheme appropriate for this latter type of legal argumentation can be set up as follows.

Previous Case Premise: C_1 is a previously decided case.
Previous Ruling Premise: In case C_1, rule R was applied and produced finding F.
New Case Premise: C_2 is a new case that has not yet been decided.

Similarity Premise: C_2 is similar to C_1 in relevant respects.
Conclusion: Rule R should be applied to C_2 and produce finding F.

This scheme represents the form of argument from precedent, a central type of argument used in typical instances of legal reasoning in the common law system.

Research on retrieving arguments from legal texts that builds on these resources is already in the planning stages. A research project outlined in (Ashley 2014) plans to develop a generalized unstructured information management architecture both to recognize arguments and to construct argument trees from legal cases with the help of annotators. Unstructured information is defined as the direct product of human communication, including natural language documents, email, speech, images and video (https://www.oasis-open.org/committees/uima/charter.php). It is said to be unstructured in the sense that it lacks an explicit semantics that would enable it to be interpreted as intended by the human author. The system will use IBM's Watson QA system as well as a default logic framework based on argumentation schemes for such kinds of arguments as argument from precedent and argument from testimony about causation.

4 The Carneades Argumentation System

Another approach to the task of argument invention is that of the Carneades Argumentation System (Gordon 2010), a formal computational model of argument that is also an implemented software system. There have been four versions of the Carneades software. The first three versions were based on the formal model of argument from (Gordon et al. 2007), The source code of all four versions can be accessed on the Internet.[2] Carneades 4 is now online.[3] All four versions model arguments as directed graphs. The first implementation was a command line tool written in 2006–2008. The second (2011) implementation is a desktop application, with a graphical user interface, sometimes called the Carneades Editor. The third version was developed in the European MARKOS project (2012–2015) to build a prototype application for browsing and analyzing licensing properties of open source software.

In the version of Gordon et al. (2007), a Carneades argument graph is a bipartite, directed, labeled graph (*S, A*), consisting of a set of statement nodes, *S,* and a set of argument nodes, *A*. There are two kinds of argument nodes, pro and con. In argument maps, statement nodes are visualized with rectangles and argument nodes with circles or ovals. Pro arguments are visualized with a plus sign inside the circle, con arguments with a minus sign.

Carneades has ways to assist users with reconstructing arguments as well as ways of assisting users to construct/invent new arguments. Carneades has a tool for "validating" arguments, to check whether they properly instantiate the applied schemes. This tool can be used, for example, to find missing premises in

2 https://github.com/carneades.

3 http://carneades.fokus.fraunhofer.de/carneades.

enthymemes. The user can then take steps to fix the "invalid" arguments, or use the problems identified to pose critical questions. In addition, Carneades includes a rule engine that enables schemes to be used to fully automatically invent new arguments by applying schemes to assumptions. Carneades also has a tool designed to help users carry out this invention task.

In Carneades, argument graphs are evaluated by two factors. The first is whether or not the audience accepts the premises of the argument (Bench-Capon et al. 2007). A premise that is accepted by the audience is said to be 'in', and is shown in a green rectangular node in the user interface. A premise that is rejected by the audience is said to be 'out', it is shown in a red rectangular node in the user interface. A premise that is neither accepted nor rejected is shown in a rectangular node with a white background. The second factor is that argument weights (representing fractions between zero and one) can be assigned to each argument, indicating the strength or weakness of the audience's acceptance. Once these input values are assigned, Carneades automatically evaluates arguments by calculating whether the conclusion of a given argument should be accepted based on the audience's acceptance of the premises and on the argument link joining the premises to the conclusion. A statement which is acceptable, has been assumed to be acceptable or should be acceptable is called "in". An argument is said to be applicable if all its premises are in. Whether or not the argument link properly joins the premises to the conclusion is determined by judging whether the given argument fits the requirements of its argumentation scheme. Conflicts between pro and con arguments are resolved using proof standards, such as preponderance of the evidence or clear and convincing evidence (Gordon and Walton 2009). The preponderance of evidence standard is satisfied if the weight of the strongest pro argument is greater than the weight of the strongest con argument. Carneades has three ways in which one argument can attack and defeat another, based on the distinction of Pollock (1995, 40) between two kinds of argument attacks called rebutters and undercutters. A rebutter is an argument that attacks the conclusion of a prior argument by presenting a reason to think the conclusion is false. An undercutter attacks the argument link between the premises and the conclusion, for example by asking a critical question pointing to an exception to the holding of the argument (Verheij 2005). For example, an argument that fits the argumentation scheme for argument from expert opinion can be critically questioned by asking whether the expert is biased. The third way is to attack one or more of the premises.

The propositions accepted by the audience are input to the system. What Carneades computes is a label for each statement, where the label is "in" (green), "out" (red), or "undecided" (white). The statement is *in* if and only if it has been assumed (and not questioned), accepted by the audience or is acceptable, given its proof standard and the statement is an issue. A statement is an *issue* if it has been neither accepted nor rejected by the audience. A statement is labelled *out* if it has been assumed false (and not questioned), rejected by the audience, or if the logical complement of the statement meets the proof standard of the statement is an issue.

There are two contrasting reasoning methods commonly used in AI reasoning systems such as expert systems, an older technology that is still widely used. The typical expert system consisted of a knowledge base and an inference engine. The knowledge base stores a set of propositions taken to be factual. The inference engine applies logical rules, such as *modus ponens*, to the knowledge base and thereby infers new knowledge. This procedure can iterate as the inference engine applies rules to newly derived facts in the knowledge base to trigger yet further conclusions. Inference engines work primarily in one of two ways, called forward and backward chaining. Put in argumentation terms, they can be explained in this way. Forward chaining starts from a database representing the available facts in a case (premises) and applies inference rules (or even one, such as *modus ponens*) to them to generate conclusions which can then be used as premises to derive more conclusions. This procedure can be used to try to prove some ultimate conclusion that one wants to prove, or it can be used to simply see what can be proved. Backward chaining starts with a conclusion and works backwards to determine what premises are available in the database of the case that could be used to support this conclusion. Backward chaining starts from a designation of the conclusion to be proved, and attempts to find data that can be used to prove it. To sum up, forward chaining starts with the available data and uses inference rules to derive more data, while backward chaining starts with the conclusion and tries to find data that could furnish premises to support it.

All versions of Carneades include a rule-based inference engine for automatically inventing arguments by applying schemes to assumptions and accepted statements. The inference engine of the first three versions used backward chaining. Version 4, the latest version, uses a forward chaining rule engine (Gordon and Friedrich 2017), based on Constraint Handling Rules (Fruwirth 1991). Forward chaining rule engines have been used from the beginning of expert systems where rules are applied to propositions in a knowledge base to generate conclusions. The main advantage of this new inference engine is that it enables all argumentation schemes to be used to automatically invent arguments, including those with second-order variables (variables ranging over entire statements), such as argument from expert opinion. The earlier versions of Carneades could use such schemes for validating arguments, but not for inventing arguments.

We can begin to explain how this is related to argument invention by noting that some of the schemes, such as the one for argument from expert opinion, use a single propositional variable P as the conclusion of the scheme but it also uses other variables in the premises, such as the variable E, to stand for an expert. This second kind of variable is called a first-order variable, because it ranges over an individual, whereas the former kind of variable is called a second-order variable because it ranges over propositions. The use of second order variables, which occurs in many of the schemes in (Walton et al. 2008), is a problem for representing schemes in logic programming languages such as Prolog, as well as formal argumentation systems such as ArguMed and ASPIC+. The new formalization of version 4 of Carneades, as shown in Gordon and Friedrich (2017), has a way of handling this problem by using Constraint Handling Rules (CHR) to allow second-order variables in the premises and conclusions of schemes while generating arguments as output in a way that guarantees termination.

In this new version of Carneades each scheme is translated into a single CHR rule that can be applied to propositions in the knowledge base to infer conclusions using

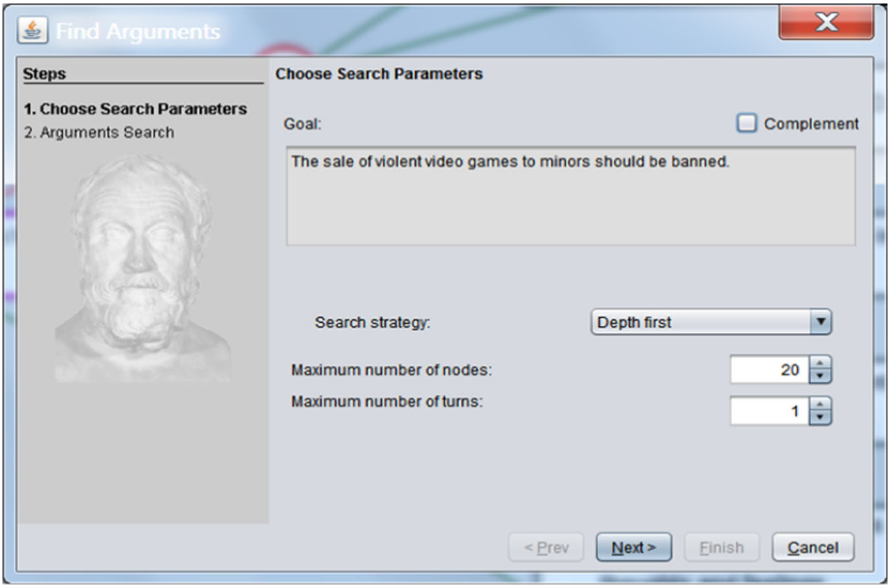


Fig. 2 The Carneades argument finding assistant

the rule. The CHR inference engine repeatedly applies the rules until no rules apply or failure is signaled. The arguments invented are then translated into Carneades arguments and added to the argument graph. Exceptions specified in the schemes instantiated by the invented arguments are used to add further arguments for undercutting the invented arguments. These arguments can then be evaluated in the usual way in Carneades. This new system has been used successfully to implement twenty of the most widely used argumentation schemes in (Walton et al. 2008), where nine of these twenty-five schemes have second-order variables (Walton and Gordon 2017).

Version 2 of Carneades has an interactive argument finding assistant, with a graphical user interface, shown in Fig. 2. The assistant helps a user to invoke the inference engine to automatically invent new arguments to support a selected claim and have the argument graph extended with these new arguments. Here version 2 is used, even though version 4 has the advantage of dealing more fully with second-order variables in schemes, because version 2 is more familiar as applied to tasks of analyzing and evaluating the most common and familiar kinds of arguments dealt with by scholars, teachers and students of argumentation theory and informal logic. Using argumentation schemes with an inference engine to invent arguments can easily produce a complex argument graph as output. Some simple as well as some more complex examples are presented in (Walton and Gordon 2012).

Using the argument finding tool, Carneades invents and puts forward new arguments and adds them to the existing argument graph (Walton and Gordon 2012). By these means, the argument graph is expanded and then re-evaluated to see if the ultimate claim is acceptable.

5 Questions About the Applicability of Argumentation Schemes

Carneades essentially uses argumentation schemes to represent an inference from a set of premises to a conclusion, and to find an argument of the type matching the scheme with a given text of discourse. To offer a simple example of how Carneades uses argumentation schemes to build a graph representing a common type of argumentation, consider a modification of the violent videogames example where each of the arguments on the two sides of the dispute has been supported by an expert source.

Let's call the two sources Dr. X and Dr. Y, both experts in psychology. Let's say that in the example, the debater on one side supports her claim that the sale of violent videogames to minors should be banned by arguing that according to an expert opinion, exposure to violent videogames results in aggression-related thoughts and feelings. The debater on the other side presents a con argument stating that it has been claimed by another expert that violence in videogames is not causally linked with aggressive tendencies.

To reconstruct this argument, the argumentation scheme for argument from expert opinion (Walton et al. 2008, 310) can be applied.

First Premise: E is an expert in domain *D*.
Second Premise: E asserts that *A* is true.
Third Premise: A is within *D*.
Conclusion: Therefore *A* may tentatively be accepted as true.

The pro argument shown at the top of Fig. 3 contains textual information that fits three of the premises of the scheme. Dr. X is an expert in psychology and he said that exposure to violent videogames results in aggression-related thoughts and feelings. These first two premises of the argumentation scheme have been explicitly stated in the text of the case given above. But there is also an implicit premise needed to meet the requirements of the scheme. This implicit premise assumes that the statement 'exposure to violent videogames results in aggression-related thoughts and feelings' fits into the domain of psychology. This is the middle premise of the pro argument shown in Fig. 3.

What the application of an argumentation-based system such as Carneades has enabled the user to do is to find an implicit premise needed to be added to the given argument. This is part of the argument reconstruction task, even though it may seem like argument invention, because what has happened is that the user has been enabled to find a missing premise in an argument.

The situation is similar in the con argument at the bottom of Fig. 3. A different implicit premise has to be added into meet the requirements of the scheme. This is that statement 'Causal linkage of violent videogames with aggressive tendencies falls into the domain of psychology'. So once these arguments are structured using the argument diagram and the argumentation scheme for argument from expert opinion as shown in Fig. 3, it is shown that the con argument rebuts the pro argument. The debate is a deadlock, but the deadlock can be broken using Carneades by assigning weights of argument strength based on audience acceptance to each of the two arguments.

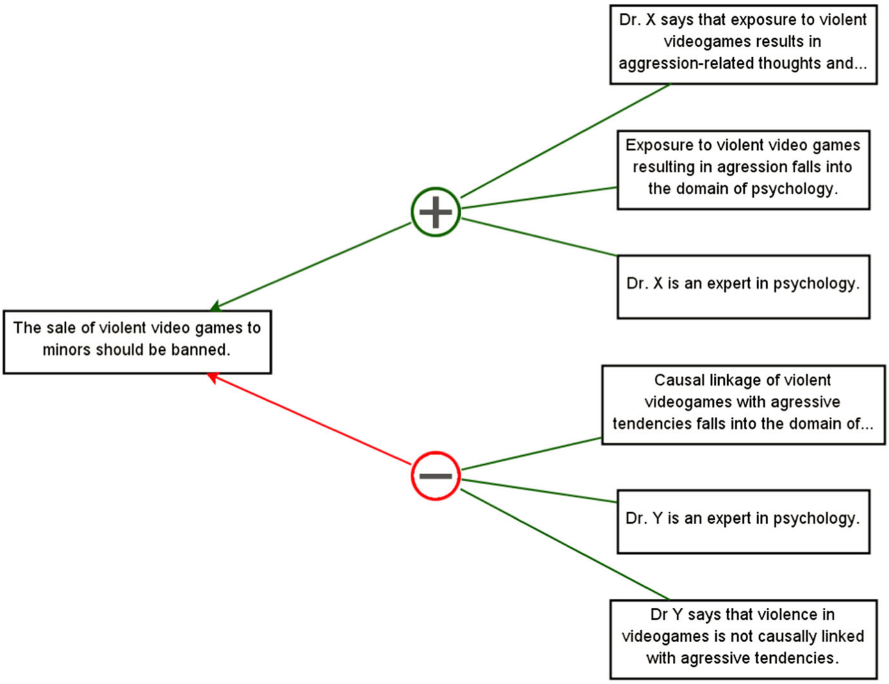


Fig. 3 Applying a scheme in constructing a Carneades argument diagram

However there is a technical problem that arises on how to represent this argument in a Carneades argument graph. The statement 'Exposure to violent videogames resulting in aggression falls into the domain of psychology' is not all that different from the statement 'Causal linkage of violent videogames with aggressive tendencies falls into the domain of psychology'. Saying that violent videogames resulted in aggression is not all that different from saying that there is a causal linkage between violent video games and aggressive tendencies. Perhaps both statements could be combined into a single premise shared by both arguments. This interpretation is expressed by the argument diagram shown in Fig. 4. It is possible in Carneades for two arguments to share a common premise, and this is the very kind of situation illustrated by Fig. 4. The difference between Figs. 3 and 4 raises the question of whether there is a textual entailment, or even a textual equivalence, between the two implicit premises shown separately in Fig. 3. What is suggested is the possibility that methods of textual entailment could be applied to this example and others like it that could form the basis of the decision on which way such an argument should be analyzed using argument diagrams.

An apparent limitation of this approach is that in many instances the type of inference represented by a given argument does not match any of the known, existing argumentation schemes. This limitation can at least partially be overcome by realizing that the scheme called defeasible *modus ponens* can be brought to bear in such cases.

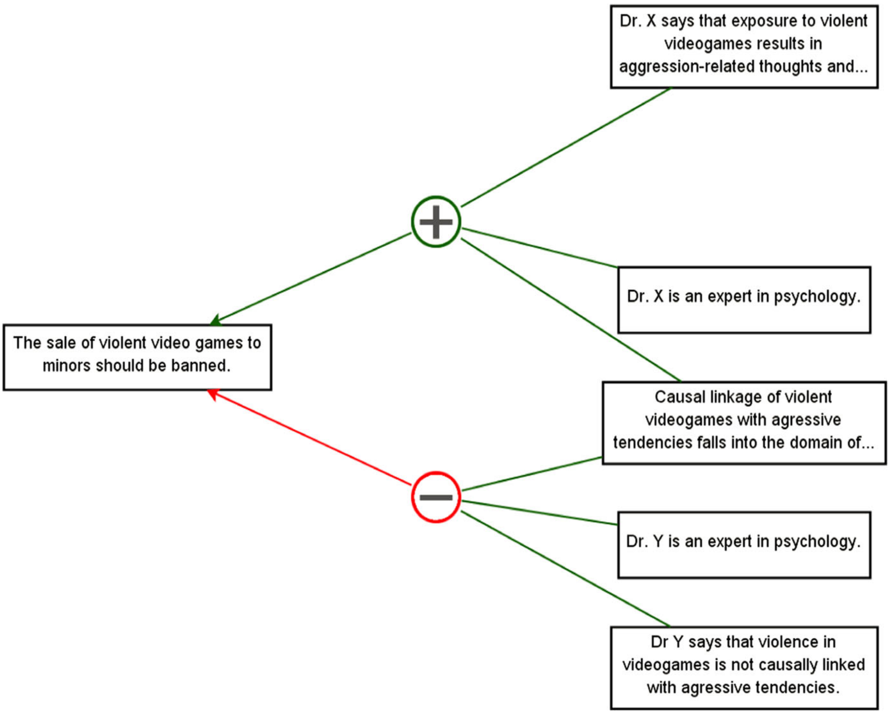


Fig. 4 Applying a scheme in constructing a Carneades argument diagram

The basic scheme for argument from expert opinion above looks like it could be expressed as a form of *modus ponens* argument resting on an additional premise that has a conditional form. This transformation can be shown by adding an implicit conditional premise to the basic argumentation scheme as follows.

Conditional Premise: If *E* is an expert in domain *D*, and *E* asserts that *A* is true and *A* is within *D*, then *A* may tentatively be accepted as true.
First Premise: *E* is an expert in domain *D*.
Second Premise: *E* asserts that *A* is true.
Third Premise: *A* is within *D*.
Conclusion: Therefore *A* may tentatively be accepted as true.

This scheme does not fit the deductively valid form of *modus ponens*, because argument from expert opinion is a defeasible (non-monotonic) form of argument. But it could fit a form of argument called defeasible *modus ponens* (DMP) by Walton (2002).

Suppose that the statement 'If *A* then defeasibly *B*' is accepted.
Suppose also that *A* is accepted.
It follows that *B* can be accepted as an assumption, subject to later retraction.

DMP is on the list of schemes implemented in version 4 of Carneades.

6 An Example to Explain How the Carneades Finds Arguments

The reader who wants real examples of Carneades argument invention can look at Gordon (2011a) or (b). The latter paper uses argument invention in complex legal cases of open source software compatibility issues. In this section we present a simple but abstract example to show in general terms how the Carneades invention system works.

Carneades uses argument diagrams (argument maps) to visualize each argument graph structure. For example, the diagram shown in Fig. 1 is an example of a Carneades argument graph. The conclusion of the argument is shown in a rectangle at the far left of the figure. The three pro arguments are shown at the top of the figure while three con arguments are shown at the bottom in this instance. So in this case, each of the six arguments has a single premise, and whether a given argument is pro or con the ultimate conclusion at the left is shown by the notation in the six circular nodes.

An example of a Carneades argument graph is shown in Fig. 5. The ultimate conclusion p0 is shown at the left. Argument a1 is shown as a pro-argument with a weight of 0.7 that supports the ultimate conclusion. Argument a2 is a con argument that attacks the ultimate conclusion and has an argument weight of 0.5. Let's assume that the standard of proof in the example is preponderance of the evidence (the default proof standard of Carneades). The preponderance standard is met by a statement if it is supported by at least one applicable pro argument which weighs more than every applicable con argument.

Let's consider the con argument a2 first. Neither of its premises, p3 and p4, has been accepted by the audience. However, p3 has an argument a5 supporting it with a weight of 0.7, and the premise p8 of a5 has been accepted by the audience, as indicated by the green color in its node.

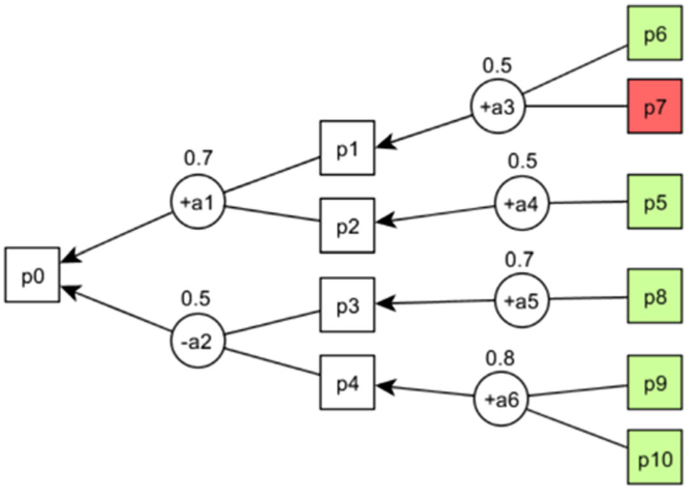


Fig. 5 First step of argument invention in the videogames example

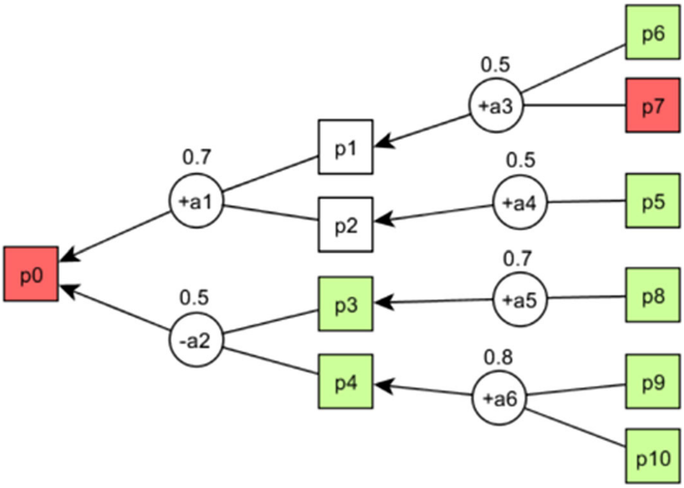


Fig. 6 Second step of argument invention in the videogames example

Thus a5 is applicable and p3 meets the preponderance standard, since there are no con arguments, and is thus acceptable. In Fig. 6, Carneades now shows p3 with a green background. The argument a6 supporting premise p4 is also applicable, since both its premises are accepted by the audience. Since p4 has no con arguments and is supported by a6, it is acceptable and now also shown with a green background. Now, as shown in Fig. 6, since a2 is a con argument, and both of its premises are acceptable, this argument is currently a successful rebuttal of the conclusion p0. Hence p0 is now shown in a red box.

Now let's look at the pro argument a1 shown at the top of Fig. 6. It has a weight of 0.7, so it could potentially defeat the con argument a2. One of its premises, p2, can now be shown as acceptable, because it is supported by a4, which has only one premise, and that premise is accepted. But that will still not help a1, because its other premise p1 is not accepted, and p1 is only supported by argument a3 that has one premise that is rejected by the audience. So far then the system has not provided a way for the advocate who wants to get the audience to accept p0.

What advice can be given by the Carneades argument assistant to remedy this situation? One piece of advice is to search for a new argument that could be used to get the audience to come to accept p7 by looking for a pro argument that supports p7. A second piece of advice would be to attack one of the premises of a5 or a6. A third would be look for arguments that would undercut a5 or a6. A fourth would be to look for a different pro argument with a weight of > 0.5 that supports p0. These four strategies are shown in Fig. 7.

In each instance the Carneades argument assistant follows the procedure in Fig. 2 to invent arguments that will fit the four strategies shown in Fig. 7. For example, the assistant might invent a pro argument that is strong enough to overcome p7's being rejected (because it has not been assumed to be true and there are no arguments supporting it) and has premises that the audience accepts. If no argument of this

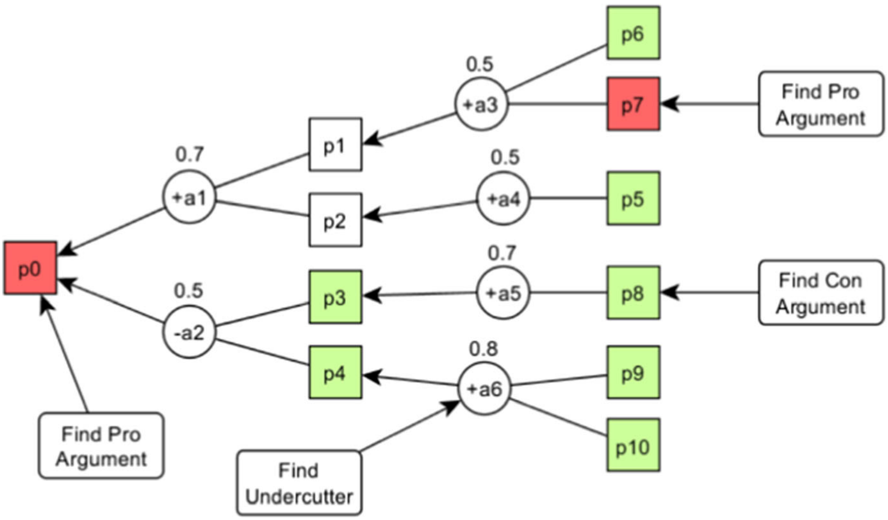


Fig. 7 Third Step of Argument Invention in the videogames example

kind can be invented using the knowledge base of argumentation schemes and the audience's commitment store, the assistant might turn up one that only lacks a premise that can supported by another argument. The search can continue by following along the general lines of the procedure displayed in Fig. 7. This is an advantage of backwards chaining as used in Carneades 2. The latest version, Carneades 4, constructs arguments by forward chaining from the statements accepted by the audience (or assumed to be accepted), and thus does not construct arguments with premises that have not been accepted.

7 Specific Points of Comparison Between Debater and Carneades

Both Debater and Carneades find a central place in their argumentation procedures for the distinction between pro and con arguments. It could be said that both systems essentially view argumentation as a technology for identifying and comparing pro and con arguments in relation to a given issue or topic (Row 1 of Table 1). Carneades invents arguments by applying argumentation schemes, using a rule-based inference engine, to a set of statements representing statements accepted, or assumed to be accepted, by the audience, while Debater searches for claims called CDC's (Row 2).

In Carneades an argument is a structure instantiating an argumentation scheme and linking a set of premises to a conclusion. In Debater, an argument is defined as a pro or con relation between a claim and the topic (Row 3). Carneades has two knowledge bases that it uses to invent arguments. One is a set of statements (propositions) representing the commitments accepted, or assumed to be accepted, by the audience. The other is a set of rules representing argumentation schemes.

Table 1 Points of contrast between Carneades and Debater

#	Carneades	Debater
1	Invents and weighs pro/con arguments on an issue	Provides pro/con arguments on a topic
2	Invents arguments by applying argumentation schemes to assumptions, using a rule-based inference engine	Searches for claims called CDC's
3	Argument is a premises-conclusion structure, instantiating an argumentation scheme	Argument is a relation between claim and topic
4	Invents arguments using a knowledge base of schemes and assumptions, represented in a formal knowledge-representation language	Searches a natural language database
5	Domain experts (knowledge engineers) represent argumentation schemes using a knowledge-representation language	Extracts CDC's from a natural language text
6	Makes use of schemes for invention	Does not make use of schemes for invention
7	Automatically classifies arguments as pro or con	Automatically classifies as pro or con
8	Designed for, but not limited to, legal argumentation	Everyday argumentation is central
9	Argument reconstruction, by interpreting natural language texts, is manual, but with tool support, using argumentation schemes	Debater uses human labelers
10	Carneades uses schemes for inferences	Debater uses textual entailment for inferences

Debater searches through a full text database such as Wikipedia where propositions representing claims have to be dug out of the text using criteria to identify CDC's and to exclude kinds of sentences not fitting the criteria for a CDC (Row 4).

Debater makes extensive use of tools from computational linguistics to extract CDC's from a natural language full text database (Row 4). Carneades assumes that the work of extracting the propositions that should go into its propositional knowledge base, representing the commitments of the audience, has already been done. Carneades makes extensive use of argumentation schemes in its argument invention procedure (Row 6). Debater does not make use of argumentation schemes in its argument invention procedure (Row 6). When Debater finds a CDC, it automatically classifies the CDC as pro or con the topic representing the ultimate claim to be proved or disputed in a given case (Row 7). When Carneades invents an argument, by using the inference engine to instantiate an argumentation scheme, it automatically classifies the argument as pro or con (Row 7). Carneades has been developed mainly to support legal argumentation, but is domain independent (Row 8). Debater takes everyday natural language argumentation as its domain of application, meaning conversational argumentation of the kind that would be found, for example, in Wikipedia entries.

A key difference between the Debater and Carneades is that Debater uses labelers to pick out a CDC by finding its boundaries in a text, based on criteria furnished to the labelers. By solving problems confronted by the labelers' attempts to identify CDC's, the research goal is to improve this selection process and make it more highly automated (Row 9). In Carneades, in contrast, knowledge engineers are needed to represent the argumentations schemes of the application domain in a rule-base. Thus, to use Carneades to automatically invent arguments, one must get through the knowledge acquisition bottleneck.

A final key difference between the two approaches concerns the operation of how inferences are drawn from statements to other statements (such as from premises to conclusions, or from a sentence in the given discourse to a statement representing its meaning) by each system. These points of contrast are summarized in Table 1.

Debater uses textual entailment to manage the operation of how inferences are drawn from statements to other statements whereas Carneades uses argumentation schemes (Row 10). According to (Dagan et al. 2006, 1), textual entailment is defined as an asymmetrical relation between a chunk of text, denoted by *T*, and an entailed hypothesis, denoted by *H*. Textual entailment is used in question answering (QA) systems to identify a text that entails an expected answer (Dagan et al. 2009, i). For example suppose the text sentence is 'Yoko Ono unveiled a bronze statue of her late husband, John Lennon, to complete the official renaming of England's Liverpool airport as Liverpool John Lennon Airport'. By textual entailment, this sentence is taken to entail the answer 'Yoko Ono is John Lennon's widow'.

More generally (Dagan et al. 2006, 1), *T* is said to entail *H* "if the meaning of *H* can be inferred from the meaning of *T*, as it would typically be interpreted by people". This definition is said to be an informal one that assumes common human understanding of language as well as common background knowledge. In other words, application of textual entailment by a computational system assumes a common knowledge database from which knowledge can be drawn and applied to the interpretation of problematic cases. A competition underway, the Pascal Recognizing Textual Entailment Challenge has taken up the challenge of building an abstract generic task of recognizing whether one text fragment can be inferred from another (Dagan et al. 2009, viii–x).

Having now discussed comparison (10) in some detail, let's also amplify comparison (9). The following sentence is cited by Levy et al. (2014, 1494) as posing a typical problem for labelers to detect the exact CDC boundaries within a longer sentence: "The argument of deprivation states that abortion is morally wrong because it deprives the fetus of a valuable future". Since the labelers are directed to mark a concise claim rather than a compound claim they would generally be inclined to identify the CDC as the sentence 'Abortion is morally wrong', that is only one part of the longer sentence. Another reason given for selecting only this single statement rather than choosing the boundaries from the word 'abortion' to include the word 'future' is that the latter choice would make the CDC include two distinct claims.

It is interesting to contrast how argumentation systems such as Carneades would approach the same problem. In Carneades it is the user who must build the argumentation tree from the given text of discourse based on his or her understanding of the meaning of the text. Users applying this method quite typically have differences of opinion on how the text of a given case should be interpreted and put into a graph structure with premises and conclusions joined by inferential links. But in the beginning such a user would probably approach this example by observing that the word 'because' is a premise indicator term. Hence the central focus of the analysis would be that the text as stated essentially contains an argument as its kernel: "abortion is morally wrong because it deprives the fetus of a valuable future". The part before the word 'because' is the conclusion and the part after it is the premise.

8 General Differences Between the Two Approaches

Over and above these specific differences there are four general differences in how the two systems approach the task of argument invention. First, Debater uses labelers to try to pick out a CDC using criteria furnished to them. The intent is to develop the Debater procedure to the point where this selection process becomes as highly automated as possible. On the other hand, in Carneades much more of this task of interpreting the text of discourse is left up to the user. The user has the fairly complex intellectual task, on which there can be disagreements on how to represent a given argument in a text as an argument graph. The difference is that Carneades leaves much of the process of interpretation up to the user to carry out before and during building the argument graph. But once the argument is represented as a graph, Carneades can automatically make calculations to evaluate the argument relative to what the audience supposedly accepts, and apply the argument finding assistant to the task of inventing new arguments from a knowledge base of argumentation schemes to support or attack the ultimate conclusion.

Second, the approach of Carneades is quite different, because it assumes not only that the ultimate conclusion to be proved has been identified, but also that its premises, along with any attacking arguments, have been identified and configured at the very beginning of the procedure into an argument map structure, a graph. Carneades does not attempt to apply all the kinds of tools used by Debater to represent the linguistics structure of the text containing arguments. Carneades begins with a knowledge base representing a set of argumentation schemes, and the arguer's commitment store that is a set of propositions. Of course, as indicated above, some argument mining systems also make use of argumentation schemes. But Debater does not, at its present state of development, use argumentation schemes as tools to help identify arguments.

Third, in contrast to the Debater system, Carneades assumes that the knowledge base it uses to invent arguments has already been formulated as a set of argumentation schemes. There is also another way in which the structure of Carneades is different from that of Debater. Instead of searching for context dependent claims (CDC's) to support or contest a topic, Carneades invents arguments that can be used to support or attack any proposition designated as a target. How this general procedure works is shown in Fig. 8, which clearly shows how what Carneades does is something different from argument mining.

Carneades can be used to find premises that the audience does not accept that can be combined with premises the audience does accept so that potential useful arguments to move ahead with the goal of persuading the audience can be found. This part of the task is possible if the arguer can continue the persuasion dialogue with the audience and ask questions that will be useful for extracting data to add to

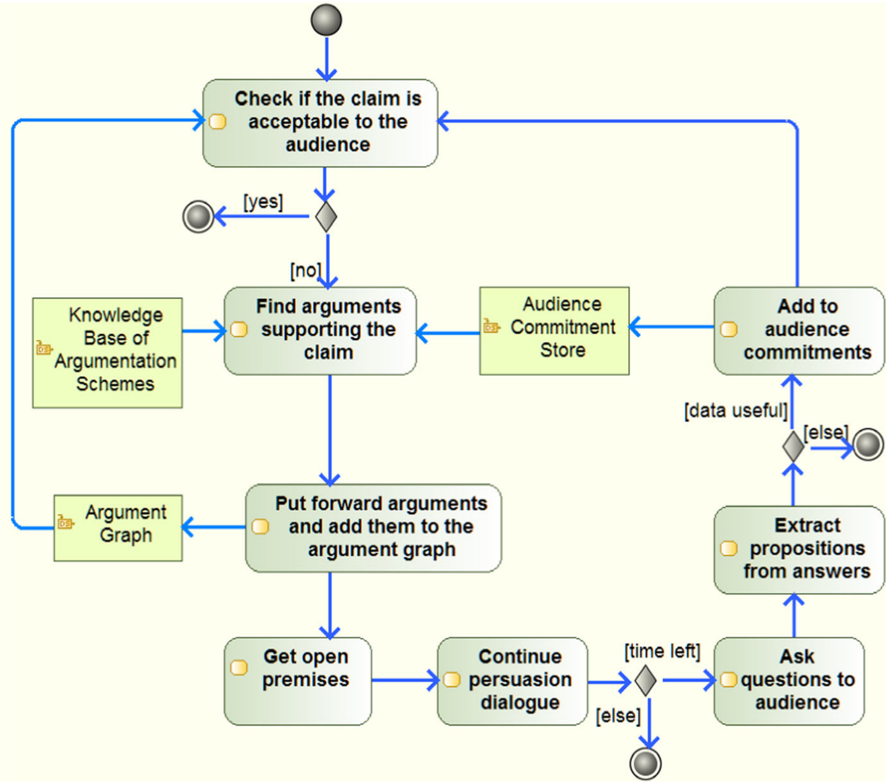


Fig. 8 An outline of how Carneades is used

the commitment store of the audience. If such useful data cannot be found, the argument invention sequence stops. But if further commitments of the audience can be turned up, these can be added to the audience commitment store and used to build new arguments that can be added to the argument graph. If these arguments can be successfully used to make the ultimate claim acceptable to the audience, the procedure stops. Otherwise it can continue further if the persuasion dialogue can be continued and turns up more useful data. So far there is no sub-module in Carneades that allows the user to perform a dialogue with the audience, but Carneades has dialogue structures as part of its formal model of a kind that could be developed in the future along the lines of the Arvina System (Lawrence et al. 2012).

A fourth point to be made is that Debater, because it does not use argumentation schemes, looks like it is meant to be a purely rhetorical system, meaning that it is designed to search for any arguments in a text that can be used to persuade the target audience to accept the designated topic proposition. Debater is an argumentation mining system in that it combs through a given text to search for any arguments that can be found in a text. These might be good arguments, or they might be bad arguments. It could be said here that such a system is *purely rhetorical*, meaning that it searches for devices of persuasion without considering whether the arguments found meet some standards for being good arguments or strong arguments, according to some normative requirements stating what makes an argument good or strong. In contrast, Carneades is a normative system that contains argumentation schemes that introduce a normative component. Debater picks out arguments from a text by polarity, and other requirements, but when it finds an argument, there is no guarantee that this argument meets normative standards. It just looks for any arguments in a text such as Wikipedia.

Debater scans through a full text database, such as Wikipedia, and finds arguments in it that have already been created by the authors of the Wikipedia articles. It is correct to say that the Debater tool does not create any new arguments that can be generalized to other domains. It only finds arguments that have already been created and can then be reused. But Carneades can be used to invent genuinely new arguments.

At first sight, the two systems might appear to be similar, and to be sure, there are some similarities. In both systems, all the end user has to put in is the designated proposition he wants to prove or refute, excluding the knowledge base of schemes (Carneades) and the full text database (Debater). Both systems are fast. Carneades typically takes less than a few seconds to invent arguments using schemes. Debater can mine relevant arguments within a few minutes. To sum up, both systems are easy and fast to use, but Carneades can require quite a bit of sophisticated input up front by a knowledge engineer, to represent the relevant argumentation schemes using a rule-based knowledge representation language, before it can automatically invent new arguments to support a designated conclusion, based on assumptions that have been put into the system.

The fundamental difference between the two systems is that Carneades can indeed invent new arguments, using a rule engine for argumentation schemes, whereas Debater is essentially a system of argument mining. The fundamental comparison is between the kind of system that uses a rule engine to invent new arguments (Carneades) and the kind of system that uses argument mining methods for finding and reconstructing arguments in natural language texts (Debater). Carneades has been used in this paper to illustrate the rule-based approach to argument invention and Debater has been used to illustrate argument mining. Debater was chosen for this comparison, because it is the most well-known and influential argument mining system.

So now we have carefully examined how the two systems work, we can see that one is a system for argument invention that uses argumentation schemes along with a rule engine to construct new arguments while the other uses argument mining methods to find argument in a natural language full text database such as Wikipedia. These observations suggest that the two systems could greatly be benefited by being combined, or by borrowing services from each other.

9 Reconsidering the Debater Example

Combining the two systems is one option to be considered. Some specific steps toward combining the two systems can be graphically illustrated by reconsidering Fig. 1. Having seen how Carneades works, the reader is now in a position to appreciate that the example chosen by the Debater system to illustrate its capabilities has actually been presented in Fig. 1 in the form of a graph, and in fact it fits the structure of a Carneades argument map. Once presented in this way, once the reader has now become familiar with how such argument maps are used in Carneades, he or she will immediately question whether some of the arguments in the map are independent from each other.

For example, look at the first con argument in Fig. 1, which states that it has been claimed that violence in videogames is not causally linked with aggressive tendencies. This con argument appears to be opposed to the middle pro argument in the three pro arguments at the top of Fig. 1, the argument stating that in addition, violent video games actually cause adolescents to commit acts of real-life aggression. In Carneades, the con argument stating that it is been claimed that violence in videogames is not causally linked with aggressive tendencies could be mapped as being a con argument against the pro-argument stating that violent video games actually cause adolescents to commit acts of real-life aggression. Moreover this same con argument stating that it has been claimed violence in videogames is not causally linked with aggressive tendencies could also be represented as a con argument against the top pro-argument stating that exposure to violent video games results in aggression-related thoughts and feelings. These two new con arguments are shown in Fig. 9.

Some might say that the top con argument in the bottom three, stating that it has been claimed that violence in videogames is not causally linked with aggressive

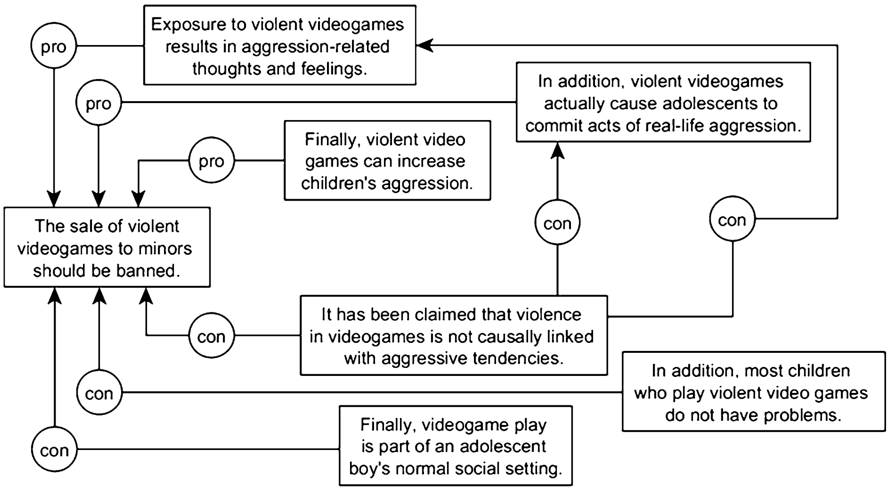


Fig. 9 Amplified argument map of the Debater example on the videogames topic

tendencies, is in fact just the opposite claim to the middle pro argument in the top three pro arguments stating that violent video games actually cause adolescents to commit acts of real-life aggression. This would not be accurate, however. It is true that the bottom CDC claims that there is no causal link, whereas the top CDC claims that violent video games actually cause adolescents to commit acts of real-life aggression. And so it appears that one is making a causal claim that the other is denying. The word 'cause' is in fact used in both CDCs. Nevertheless the bottom statement says that *it has been claimed that* violence in videogames is not causally linked with aggressive tendencies. What this suggests to the user is that Debater has found some claim, presumably made by some source, that violence in videogames is not causally linked with aggressive tendencies. Debater has searched around in its full text database, Wikipedia, and found some claim of this sort, and presumably this claim was made by some expert source, perhaps for example there have been some scientific studies that have shown using statistical evidence that there is no causal linkage between violence in videogames and aggressive tendencies in adolescents.

What these observations reveal is also very significant for comparing how Carneades could be used to analyze an argument of this sort and produce an argument map representing it. The con argument that it has been claimed that violence in videogames is not causally related with aggressive tendencies could be represented as an enthymeme, an argument with an unstated premise. For example the missing premise might state something to the effect that this claim has been made by experts, perhaps experts in psychology, who have studied this matter experimentally. The user of the Debater system could find out by putting in the statement that it is been claimed that violence in videogames is not causally linked with aggressive tendencies as a topic, and then having Debater carry out a search for pro arguments supporting this topic. How Carneades would approach the problem is different, however. Carneades could apply the argumentation scheme for argument from expert opinion, and ask the critical question 'What is the name of the expert?', and perhaps also the critical question 'What is the expert's field of expertise?' By this means Carneades would seek out additional assumptions in the form of premises needed to make the given arguments conform to the argumentation scheme for argument from expert opinion.

These observations suggest that the two systems could ultimately be combined, or used in tandem. They move toward each other by combining their virtues. Debater could begin at one end by using argumentation schemes while Carneades could be extended by merging it with argument mining tools.

10 Conclusions

In the introduction a distinction was drawn between finding and inventing. On the basis of the comparisons made in this paper, we conclude that only Carneades should be properly said to invent arguments. On our view, Debater performs the task of finding arguments that are already there in a given text or database (argument mining), and only Carneades carries out argument invention the way this task was originally described in the ancient world, and notably by Aristotle. At any rate, the underlying differences between the two systems have been explained and shown to be significant for potential users in the fields of rhetoric, informal logic and argumentation regardless of how 'finding' and 'inventing' may be defined. We conclude that both systems could lead to even more powerful tools for argument invention if their best features are combined.

A conclusion of general significance for rhetoric as field is that the uses of Carneades and Debater as argumentation invention tools are not only restricted to rhetorical persuasion. They can also be used in cases of deliberation where the goal is for a group to move forward collaboratively to arrive at a decision on what to do in a case where a choice is required. Finding arguments to support a claim that one choice is better than another, based on the known circumstances of the case and the values of the group making the deliberation, is a task that is specifically meant be carried out by version 4 of Carneades. Version 2 is about the task of evaluating or inventing arguments meant to persuade an audience that a contested thesis is acceptable or not (proving whether a designated proposition is true or false). Version 4 is about the task of evaluating or inventing arguments meant to move a deliberation forward by concluding to the best course of action. With version 4 there can be multiple issues that change as the argumentation proceeds. On this view, rhetoric applies to deliberation as well as to persuasion.

Another conclusion that appears from the examples studied in this paper is that Debater could be improved if it made use of tools for representing and evaluating structured arguments, and for inventing arguments using formal representations of argumentation schemes. These tools are already available and being used in formal and computational argumentation systems such as Carneades, among other, similar systems, such as ABA (Toni 2014), ASPIC+ (Prakken 2010), and DefLog (Verheij 2008). In particular, the use of argumentation schemes in an argument invention engine could turn out to be a valuable asset for refining Debater. Some ways of combining the argumentation approach with that of Debater have been already proposed by Cabrio and Villata (2012). Systems for argument mining from natural language texts, like IBM's Debater system, have great potential and complement existing models and tools with new functionality, but could be improved by integration with computational models of argument to reach their full potential. This path is already being taken by the recent fast developing research in computational linguistics (Peldszus and Stede 2013).

In particular, the prospects for building systems of argument invention for legal applications (Gordon 2011b) have been shown to be highly promising. These prospects could be enhanced by combining the natural language processing capabilities exhibited by the IBM Debater system with the formal argumentation systems that have been developed in existing computational systems for legal argumentation such as ASPIC+ (Prakken 2010), DefLog (Verheij 2008) and Carneades (Gordon 2010). Even if many might prefer to avoid using IBM's proprietary tools, the framing of the tools on the website is instructive in suggesting applications to legal argumentation (Ashley 2017, 41). It shows how the new text analytic technologies can be packaged in a way to make them attractive to users. These suggestions follow the current trend of concentrating less on different abstract theories of formal argumentation and more on the practical side of applying formal argumentation systems to texts of discourse containing natural language argumentation.

References

Aharoni, E., C. Alzate, A. Bar-Haim, Y. Bilu, L. Dankin, I. Eiron, D. Hershkovich, S. Hummerl, M. Khapra, T. Lavee, R. Levy, P. Matchen, A. Polnarov, V. Raykar, R. Rinott, A. Saha, N. Zwerdling, D. Konopnicki, D. Gutfreund, and N. Slonim 2014. Claims on Demand—An Initial Demonstration of a System for Automatic Detection and Polarity Identification of Context Dependent Claims in Massive Corpora. In *Proceedings of the 25th International Conference on Computational Linguistics (COLING 2014),* Dublin, Ireland, 6–9. Available as of 2014/12/17 at: http://anthology.aclweb.org/C/C14/C14-2002.pdf.

Anon. 2014. *Proceedings of the First Workshop on Argumentation Mining ACL 2014. Stroudsburg, Pennsylvania.* Association for Computational Linguistics, 2014. Available at: http://acl2014.org/acl2014/W14-21/index.html.

Anon. 2015. *Proceedings of the Second Workshop on Argumentation Mining, June 4, 2015, Denver, Colorado.* Association for Computational Linguistics, 2015. Available at: https://aclweb.org/anthology/W/W15/W15-05.pdf.

Aristotle. 1991. *On Rhetoric: A Theory of Civil Discourse,* ed. G.A. Kennedy. Oxford: Oxford University Press.

Arnauld, A. 1662. *La Logique, ou l'Art de Penser* [Commonly called *the Port-Royal Logic*]. Translated into English as *The Art of Thinking*, Dickoff, J. and James, P., 1964. Indianapolis: Bobbs-Merrill.

Ashley, K.D. (2014). *Automatically Extracting Arguments from Legal Texts: A Position Paper*. Paper Presented at the Met-Arg Workshop, JURIX 2014.

Ashley, K. 2017. *Artificial Intelligence and Legal Analytics: New Tools for Law Practice in the Digital Age*. Cambridge: Cambridge University Press.

Bench-Capon, Trevor J.M., Sylvie Doutre, and Paul E. Dunne. 2007. Audiences in Argumentation Frameworks. *Artificial Intelligence* 171 (1): 42–71.

Cabrio, E., and S. Villata. 2012, Combining Textual Entailment and Argumentation Theory for Supporting Online Debate Interactions. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics, Association for Computational Linguistics*, 208–212.

Cicero, M.T. 1949. *De Inventione* (Loeb Library Edition). Cambridge, MA: Harvard University Press.

Dagan, I., O. Glickman, and B. Magnini. 2006. The PASCAL Recognizing Textual Entailment Challenge. http://eprints.pascal-network.org/archive/00001298/01/dagan_et_al_rte05.pdf.

Dagan, I., B. Dolan, B. Magnini, and D. Roth. 2009. Recognizing Textual Entailment: Rational Evaluation and Approaches. *Natural Language Engineering* 15(4), i–xvii.

Fruwirth, T. 2009. *Constraint Handling Rules*. Cambridge: Cambridge University Press.

Gordon, T.F. 2010. The Carneades Argumentation Support System. In *Dialectics, dialogue and argumentation*, ed. C. Reed and C.W. Tindale, 145–156. London: College Publications.

Gordon, T.F. 2011a. Combining Rules and Ontologies with Carneades. In *Proceedings of the 5th International RuleML2011@BRF Challenge*, *CEUR Workshop Proceedings*103–110.

Gordon, T.F. 2011b. Analyzing Open Source License Compatibility Issues with Carneades. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Law (ICAIL-2011),* 50–55. New York, NY: ACM Press.

Gordon, T.F., H. Prakken, and D. Walton. 2007. The Carneades Model of Argument and Burden of Proof. *Artificial Intelligence* 171(10–15), 875–896.

Gordon, T.F., and D. Walton. 2009. Proof Burdens and Standards. In *Argumentation and Artificial Intelligence*, ed. Iyad Rahwan and Guillermo Simari, 239–260. Berlin: Springer.

Gordon, T.F., and H. Friedrich. 2017. *Representing Argumentation Schemes with Constraint Handling Rules, CMNA 2017*.

Kennedy, G. 1963. *The Art of Persuasion in Greece*. London: Routledge.

Kienpointner, M. 1997. On the Art of Finding Arguments: What Ancient and Modern Masters of Invention Have to Tell Us About the *Ars Inveniendi. Argumentation* 11(2): 225–236.

Lawrence, J., Bex, F., and C. Reed. 2012. Dialogues on the Argument Web: Mixed Initiative Argumentation with Arvina. In *Proceedings of the 4th International Conference on Computational Models of Argument (COMMA 2012)*, 513–514. Amsterdam: IOS Press.

Levy, R., Y. Bilu, D. Hershcovich, E. Aharoni, and M. Slonim. 2014. Context Dependent Claim Detection, *Proceedings of the 25th International Conference on Computational Linguistics (COLING 2014), Dublin, Ireland*, 1489–1500. Available as of 2014/12/17 at: http://www.aclweb.org/anthology/C/C14/C14-1141.pdf.

Mochales, R., and M.-F. Moens. (2011). Argumentation Mining. *Artificial Intelligence and Law* 19(1), 1–22.

Peldszus, A., and M. Stede. 2013. From Argument Diagrams to Argumentation Mining in Texts. *International Journal of Cognitive Informatics and Natural Intelligence* 7(1), 1–31.

Pollock, J.L. 1995. *Cognitive Carpentry*. Cambridge, MA: The MIT Press.

Prakken, H. 2010. An Abstract Framework for Argumentation with Structured Arguments. *Argument & Computation* 1(2), 93–124.

Taboada, M., and W.C. Mann (2006). Rhetorical Structure Theory: Looking Back and Moving Ahead. *Discourse Studies* 8(3), 423–459.

Toni, F. 2014. A Tutorial on Assumption-Based Argumentation. *Argument and Computation* 5(1), 89–117.

Turney, P. 2002. Thumbs Up or Thumbs Down? Semantic Orientation Applied to Unsupervised Classification of Reviews. In *Proceedings of the Association for Computational Linguistics*, 417–424.

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

Verheij, B. 2008. About the Logical Relations Between Cases and Rules. In *Legal Knowledge and Information Systems. JURIX 2008: The Twenty-First Annual Conference,* ed. E. Francesconi, G. Sartor, and D. Tiscornia, 21–32. Amsterdam: IOS Press.

Walton, D. 1994. Begging the Question as a Pragmatic Fallacy. *Synthese* 100: 95–131.

Walton, D. 2002. Are Some *Modus Ponens* Arguments Deductively Invalid? *Informal Logic*, 22(1): 19–46.

Walton, D., C. Reed, and F. Macagno. 2008. *Argumentation Schemes*. Cambridge: Cambridge University Press.

Walton, D., and T.F. Gordon. 2012. The Carneades Model of Argument Invention. *Pragmatics and Cognition* 20(1): 1–31.

Walton, D. and T.F. Gordon. 2017. Argument Invention with the Carneades Argumentation System. *ScriptED: A Journal of Law, Technology and Society* 14(2): 168–207.