dARe – Using Argumentation to Explain Conclusions from a Controlled Natural Language Knowledge Base

Adam Wyner[1] and Hannes Strass[2]

[1]Department of Computing Science, University of Aberdeen, United Kingdom
azwyner@abdn.ac.uk
[2]Computer Science Institute, Leizig University, Germany
strass@informatik.uni-leipzig.de

Abstract. We present an approach to reasoning with knowledge bases comprised of strict and defeasible rules over literals. A controlled natural language is proposed as a human/machine interface to facilitate the specification of knowledge and verbalisation of results. Techniques from formal argumentation theory are employed to justify conclusions of the approach; this aims at facilitating human acceptance of computed answers.

1 Introduction

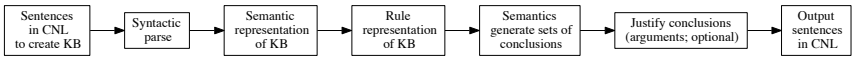
Approaches to artificial intelligence in general and to automated problem solving in particular should be – in virtue of their intelligence – able to explain and justify their conclusions and actions in a rational discourse. This is not always done: the Go playing computer program *AlphaGo* [22], while very proficient in choosing the right move (i.e. solving a range of problems), cannot explain to a human user *why* it chose that particular move (i.e. justifying its solution). A recent Nature editorial concluded that "[t]he machine becomes an oracle; its pronouncements have to be believed." (Nature 529, p. 437)

To make believable, useful results, they have to be communicated to human users, which implies that the formal knowledge models used in efficient inference mechanisms ought to be translatable into a form that is familiar and relevant for humans. In this paper, we aim at addressing specific problems of usability of knowledge-based intelligent systems in a particular, restricted setting. The restricted setting is that of reasoning with non-monotonic semantics of knowledge bases (KBs) that are given in the form of strict and defeasible rules, since people reason non-monotonically about many matters. For this, we make use of several techniques. Firstly, to address the communication issue (between humans and machines), we employ a *controlled natural language* as specification language for the input of the model as well as the output of inferences. Controlled natural languages (CNLs) are subsets of natural language that have been restricted in lexicon and grammar, thereby eliminating ambiguity and reducing complexity [17]. Some systems automatically translate sentences into formal, machine-readable semantic representations; they are useful in, for example, conversational intelligence analysis, so support distributed sense-making [25]. We adapt one such system, AceRules [16], for user specification of defeasible theories. Secondly, to address the explanation issue (justifying answers), we employ techniques from formal argumentation theory. Argumentation studies determine which arguments are acceptable, that is, which arguments

can be defended in rational discourse, where arguments consist of prerequisites, a claim, and an inference between the two, along with their relationships with other arguments, such as rebuttal. Formal argumentation theory and its implementations formally and automatically construct conclusions from a knowledge base. The CNL interface allows a user to build the knowledge base and to receive justified conclusions in natural language. Importantly, as we argue, a CNL enables an *engineering approach* to argumentation and reasoning in natural language. This is in contrast to most existing approaches to formal argumentation, which do not strongly tie-in to intuitions about natural language. It also contrasts with argument mining [18], which, while promising, requires extensive preprocessing and normalisation to support formal inference.

On the reasoning side, there are approaches to reasoning with knowledge bases consisting of strict and defeasible rules [6,10,7,21,1,26,23,2,8]. We subscribe to none in particular and opt to make our approach parametric, abstracting from the concrete reasoning back-end that is used. We only assume that the back-end receives as input a set of strict and defeasible rules, provides output in the form of conclusions with respect to some semantics, and yields (upon request) justifications of the conclusions.

The novel contributions of this paper are that we propose a new interface between natural language, defeasible knowledge bases, and defeasible reasoning, which is either non-existent in other approaches or does not correlate with intuitive semantic judgements. Our proposal is the first to facilitate automatic reasoning from inconsistent knowledge bases in natural language [16,12,13]. It takes the idea of "convincing by explaining" one step further, as explanations can be expressed in a natural language text. We can apply and propose to extend an existing controlled natural language (CNL) tool that largely provides the requisite translation to defeasible theories, which can be further processed by background reasoning engines. We dub our system *dARe* for "*d*efeasible *A*c*e*-*R*ules with *e*xplanations". In the rest of the paper, we provide a motivating example, introduce the formal language on which the actual reasoning is done, then outline our natural language interface. The picture below illustrates the overall process which is discussed over the course of the paper. Our aim is to provide a high level outline of the issues relating to defeasible reasoning and CNLs; a detailed or formal presentation is beyond the scope of this work and can be better appreciated from the references. We close with some discussion and notes on future work.



2 A Motivating Example

For human-machine communication, there are CNL tools which translate natural language into first-order logic formulas and interface to (non-monotonic) inference engines [16,12,11,13]. Yet, there are still issues with defeasible and/or conflicting information. More pointedly, defeasible propositions are modelled using "not provably not", which we show has a different interpretation than the natural expression "usually" or "it is

usual that", which are normative quantifier expressions over contexts [15]. The following running example is paraphrased from Pollock [20] and illustrates these matters.

Example 1 (Moustache Murder).

> *Jones is a person. Paul is a person. Jacob is a person. Usually, a person is reliable. If Jones is reliable then the gunman has a moustache. If Paul is reliable then Jones is not reliable. If Jacob is reliable then Jones is reliable.*

Clearly not both Paul and Jacob can be reliable. Crucially, any semantics should provide a choice between the different (and mutually exclusive) consistent viewpoints of this narrative. An interpretation of "usually" should facilitate such choices.

In the approaches of [11] and [13], the adverb of quantification "usually" is translated as "not provably not" (perhaps along with an abnormality predicate), e.g. a paraphrase for "usually, a person is reliable" is along the lines of "if a person is not provably not reliable then the person is reliable". However, this formalisation can be incorrect, as demonstrated by its straightforward ASP implementation:

```
1: person(jones). person(paul). person(jacob).
2: has(gunman,moustache) :- reliable(jones).
3: -reliable(jones) :- reliable(paul).
4: reliable(jones) :- reliable(jacob).
5: reliable(X) :- person(X), not -reliable(X).
```

This answer set program is inconsistent. The literal `-reliable(jacob)` cannot ever be derived from the program, so `reliable(jacob)` must be in every answer set by `(5)` and `(1)`. Thus `reliable(jones)` must be in every answer set by `(4)`. However, the same holds for `paul`, whence the literal `reliable(paul)` must be in every answer set. Thus `-reliable(jones)` must be in every answer set by `(3)`. Consequently, any answer set would have to contain both `reliable(jones)` and `-reliable(jones)`, therefore no answer set exists.[1] Yet, a correctly formalized logic program ought to produce the intended interpretations as stable models. Thus, the "not provably not" reading of "usually, $\langle statement \rangle$" phrases is not always correct.[2] In contrast, the correct reading is obtained by interpreting "usually, $\langle statement \rangle$" as a defeasible rule in a defeasible theory. In the following section, we outline our approach to defeasible theories.

3 Defeasible Theories

For a set $\mathcal{P}$ of atomic propositions, the set $\mathcal{L}_\mathcal{P}$ of its literals is $\mathcal{L}_\mathcal{P} = \mathcal{P} \cup \{\neg p \mid p \in \mathcal{P}\}$. A *rule* over $\mathcal{L}_\mathcal{P}$ is a pair (B, h) where the finite set $B \subseteq \mathcal{L}_\mathcal{P}$ is called the *body* (premises) and the literal $h \in \mathcal{L}_\mathcal{P}$ is called the *head* (conclusion). For $B = \{b_1, \ldots, b_k\}$ with $k \in \mathbb{N}$, we can write rules thus: a *strict* rule is of the form "$b_1, \ldots, b_k \to h$"; a *defeasible* rule is of the form "$b_1, \ldots, b_k \Rightarrow h$". In case $k = 0$ we call "$\to h$" a *fact* and "$\Rightarrow h$"

[1] While ASP can deal with this example, the common "not provably not" reading of "usually, $\langle statement \rangle$" phrases is not always correct.

[2] Adding an abnormality atom into the body of line 5 (like in rule (12) of [5]) would address inconsistency, but not get us our intended reading. It would introduce the issue of having to create abnormality predicates from language input, where such predicates are not explicit.

an *assumption*. The intuitive meaning of a rule (B, h) is that whenever we are in a state of affairs where all literals in B hold, then also literal h (always/usually, depending on the type of rule) holds. A *defeasible theory* is a tuple $\mathcal{T} = (\mathcal{P}, \mathcal{S}, \mathcal{D})$ where $\mathcal{P}$ is a set of atomic propositions, $\mathcal{S}$ is a set of strict rules over $\mathcal{L}_\mathcal{P}$, and $\mathcal{D}$ is a set of defeasible rules over $\mathcal{L}_\mathcal{P}$. In this paper, we will also consider defeasible theories with first-order predicates, variables, and constants, and treat them as short-hand versions of their ground instantiations. More details can be found in a previous workshop paper [24].

The semantics of defeasible theories are a topic of ongoing work in argumentation theory [6,10,7,21,1,26,23,2,8]. For the purposes of this paper, we express no preference, abstracting away from any concrete manifestations of existing approaches. For our approach to work, we make a few (mild) assumptions about the approach to assigning semantics to defeasible theories that is used to draw inferences (the "back-end"). More specifically, we assume that the reasoning back-end:

1. ...accepts a defeasible theory $\mathcal{T} = (\mathcal{P}, \mathcal{S}, \mathcal{D})$ as input. We consider this a mild assumption since only in some cases an additional step might be needed to transform $\mathcal{T}$ into the reasoner's native input format. (Some approaches distinguish rules with empty and non-empty bodies [10,21,2], which can be achieved by a simple syntactic preprocessing step; in ABA [6], there are no defeasible rules with non-empty body, this can be checked before passing the theory to the reasoner.)
2. ...can produce "interpretations" (consistent viewpoints, e.g. extensions) and/or (sets of) credulous/sceptical conclusions of the defeasible theory with respect to one or more semantics, e.g. stable, complete, preferred, grounded [9].
3. ...can produce graph-based justifications for its conclusions. For most approaches, this will be easy as they use structured (mostly tree-shaped) arguments, and when queried for the justification for a single conclusion, can just return a derivation of that literal as obtained from an argument extension. We assume only graph-based justifications in our approach to be most general, as more recent approaches [8,24] diverge from the traditional tree-shaped view for computational reasons.

It may be more or less straightforward to lift these restrictions, depending on the concrete approaches. Our assumptions cover considerable common ground of the various approaches in the literature; they are a meaningful and non-trivial starting point for our own work. While there are several roles for argumentation, for our purposes, it serves to provide graph-based justifications for conclusions, which contrasts to other approaches. We illustrate the formal back-end language with our running example.

Example 1 (Continued). The text on the gunman mystery leads to the below defeasible theory with variables, where Π is a set of (first-order) predicates, $\mathcal{C}$ is a set of constant symbols, $\mathcal{V}$ is a set of variables, $\mathcal{S}$ is a set of strict rules, $\mathcal{D}$ is a set of defeasible rules, and $\mathcal{T}$ a theory constructed from the other components (using atoms over Π, $\mathcal{C}$, and $\mathcal{V}$):

$$\Pi = \{person/1, reliable/1, has/2\}\,,\ \mathcal{C} = \{jones, paul, jacob, gunman, moustache\}$$
$$\mathcal{T} = (atoms(\Pi, \mathcal{V}, \mathcal{C}), \mathcal{S}, \mathcal{D}) \text{ with } \mathcal{D} = \{person(x_1) \Rightarrow reliable(x_1)\}, \text{ and}$$
$$\begin{aligned}\mathcal{S} = \{&\rightarrow person(jones), \quad \rightarrow person(paul), \quad \rightarrow person(jacob),\\ &reliable(jones) \rightarrow has(gunman, moustache),\\ &reliable(paul) \rightarrow \neg reliable(jones), \quad reliable(jacob) \rightarrow reliable(jones)\}.\end{aligned}$$

Intuitively, this defeasible theory ought to have two different "interpretations":

$M_1 = M \cup \{reliable(jacob), reliable(jones), has(gunman, moustache)\}$ and
$M_2 = M \cup \{reliable(paul), \neg reliable(jones)\}$, with
$M = \{person(jones), person(paul), person(jacob)\}$.

In particular, either of these two sets makes a choice whether Jacob is reliable or Paul is reliable, avoiding inconsistency.

4 Obtaining Defeasible Theories from Controlled Natural Language

In the subsections below, we justify CNLs for argumentation and knowledge bases, outline an existing CNL and required modifications, and then discuss our running example. Our aim is to provide a high level outline of the issues, as a detailed or formal presentation of a CNL and associated inference engine is beyond the scope of this work and can be better appreciated from the cited literature.

4.1 The Role of a CNL in Argumentation

We claim that a CNL is an important, perhaps essential, interface to argumentation, wherein natural language input is automatically analysed (parsed and semantically represented), returning a formal representation suitable for reasoning; the results are then verbalised in natural language. Our proposal is the first to facilitate automatic reasoning from inconsistent knowledge bases in natural language [16,12,13].

Our approach is complementary to argument mining, where texts are extracted from unstructured natural language corpora, then mapped to arguments for reasoning in Dungian AFs [18], given some sense of what counts as an argument [26]. In current argument mining approaches, machine learning techniques are often applied to identify topics, classify statements as claim or justification, or relate contrasting statements. However, natural language is highly complex and diverse in lexicon, syntax, semantics, and pragmatics. Current mining approaches do not systematically address matters of synonymy, contradiction, or deductions. They treat extracts from texts as atomic propositions, so do not provide *fine-grained* analysis into a formal language suitable for knowledge representation and reasoning such as predicate logic, where predicates and individuals are articulated (also see the *recognizing textual entailment* tasks [3]). Therefore, it is difficult to account for a range of patterns of reasoning that are fundamental to argumentation in natural language.

In contrast to argument mining, we adopt a CNL-based approach, where a CNL is an engineered language that reads as a natural language, yet has a constrained lexicon and grammar [17]. We are particularly interested in CNLs which translate the input language to machine-readable, first-order logic expressions and which interface with inference engines for model generation and theorem proving. Such a CNL facilitates an *engineered solution* to argumentation in NL by addressing three critical issues.

First, it provides *normalised language* which, in principle, can serve as target expressions for information extracted by argument mining; thus we can process arguments and reason in the requisite way. For example, a CNL can homogenise diverse linguistic forms, e.g. passive and active sentences, and disambiguate expressions using interpretation rules. Second, we can *scope, experimentally control, and systematically augment the language* as needed. Finally, ACE gives us an *essential experimental interface with inference engines*, enabling testing of different forms and combinations of transformations from natural language to a formal language, then the interaction with alternative inference engines. Thus, in our view, a CNL is not only compatible with approaches to argument mining, but arguably a prerequisite processing pipeline element to instantiate substantive and articulated knowledge bases that facilitate inference.

While there are, in our view, clearly advantages to working with a CNL, it is important to acknowledge its limitations as well. As an engineered language, there are lexical items and grammatical constructions that are not available from the source natural language. Relatedly, there are interpretive, contextual, or idiomatic matters that a CNL does not address. Users of a language must learn to work with a CNL in ways that they do not in natural languages. Despite these limitations, we believe the advantages of an engineered language outweigh them.

4.2 AceRules

We work with AceRules [16], which is a sublanguage of Attempto Controlled English (ACE)[3] [12,16] (also see RACE [12], PENG-ASP [13], and ITA Controlled English [19]). For our purposes, the main advantage of AceRules over ACE is that AceRules allows us to access and redirect the inference engine, facilitating comparison between existing proposals and our alternative proposal. While many functionalities of AceRules are currently available and useful, other key components have been identified as to be implemented and are as of yet manually produced.

AceRules has a range of lexical components: proper names, common nouns, logical connectives, existential and universal quantifiers, one and two place predicates, and relative clauses. Construction rules define the admissible sentence structures, e.g. declarative or conditional sentences. The admissible sentences are translated into *Discourse Representation Structures* [14,4], which can be translated into predicate logic and which support the semantic representation of aspects of discourse such as pronominal anaphora. For instance, a sentence such as "*Every man is happy.*" is automatically parsed and semantically represented along the lines of $\forall x[man(x) \rightarrow happy(x)]$. Interpretation rules restrict input such that each sentence is provided a single, unambiguous translation into a semantic representation; a user must evaluate whether this representation is the intended interpretation. There are further lexical elements and syntactic constructions to use as needed. Verbalisation generates natural language expressions from the formal representations, which fulfils the basic objective of making the results of inference accessible. A range of auxiliary axioms (from ACE) can be optionally added to treat generic linguistic inferences, e.g. interpretations of "be", relations between the plural and the singular form of nouns, lexical semantic inferences such as *throw* implying

[3] `http://attempto.ifi.uzh.ch/site/description/`

move, and a range of presuppositions relating to proper names or definite descriptions. Domain knowledge must be added as well into AceRules. To use AceRules, the user has to have some familiarity with the vocabulary, grammar, and interpretation rules. There are a range of support tools to input statements correctly, represent them in different forms, and process them further such as for reasoning or information extraction.[4]

For the semantics, AceRules has linguistic expressions and correlated first-order representations for strong negation, negation-as-failure, the strict conditional, and the adverb "usually", which is a predicate of events. It connects to different inference engines (courteous logic programs, stable models, and stable models with strong negation) and allows others, e.g. our direct semantics from [24].[5] These features are sufficient to reason non-monotonically. However, there are two key problems with AceRules as is (and shared with RACE and PENG-ASP): it cannot reason from inconsistent knowledge bases (e.g. as in the Nixon diamond example), and it does not incorporate the defeasible conditional. We have argued that both are essential for human-like reasoning. We have shown (see Example 1) that a conditional with "not provably not" is not semantically equivalent to the natural interpretation of "usually $\langle statement \rangle$" as the defeasible conditional. More relevant to the discussion of a natural language interface to arguments to explain conclusions, AceRules does not parse, semantically represent, or verbalise the discourse connectives such as "because" or "except", which are essential for natural expressions of explanation and justification (neither does ACE).

To address these matters, it is necessary to modify AceRules in several ways. Some of the modifications are as yet to be integrated for automatic processing. AceRules allows modal operators as sentential modifiers, e.g. "It is possible that" as well as the modal auxiliary "may", which may only be applied to atomic sentences. There is no sentential adverb "usually", but there is a manner adverb "usually", which is a predicate of events that does not provide the intended interpretation. To avoid problematic polysemy, we have created a new lexical item "usual" that is a predicate of atomic sentences as in "It is usual that P", where P is an atomic sentence; sentences of such forms are parsed by the revised AceRules. Further revisions are under development. In particular, expressions of the form "It is usual that P" are to be semantically represented with the defeasible conditional: where we have "It is usual that P" appears as a defeasible rule without a body; where we have "If Q then it is usual that P", we have a defeasible rule with Q as the body and P as the head. In addition, the semantic representation of a knowledge base with defeasible rules is to be processed with an inference "back-end" as described in Section 3, e.g. [24].[6] Finally, as previously noted, AceRule's verbalisation must be augmented to present structured explanations for extensions, which are arguments for conclusions; that is, we would like expressions such as "P because Q." to indicate what is concluded from the knowledge base, e.g. P, along with its justification,

[4] See [27,28] for an example of several natural language statements that are worked with ACE and related to an instantiated argumentation framework.

[5] RACE and PENG-ASP have the same expressions [12,13]. RACE is based on *Satchmo* (written in Prolog), while PENG-ASP uses ASP.

[6] An integration to AceRules is feasible; see, in a related setting, *If Nixon is a quaker then Nixon usually is a pacifist.* in `https://argument-pipeline.herokuapp.com/`, which is based on [26]. However, that work relied on ad-hoc manipulations of semantic representations.

e.g. Q. In this way, we can fulfil the goal of defeasible reasoning in natural language along with justifications for conclusions.

4.3 Worked Example

In this section, we discuss our running example further, showing some of the revised linguistic forms needed to make the original example into expressions that are compatible with AceRules. The original statement from [20] is provided in the first portion below, followed by our paraphrase as given previously, and then an additional paraphrase written so as to be compatible with AceRules.

Example 1 (Continued). Moustache Murder

> *Source: Jones says that the gunman had a moustache. Paul says that Jones was looking the other way and did not see what happened. Jacob says that Jones was watching carefully and had a clear view of the gunman.*
>
> *Paraphrase 1: Jones is a person. Paul is a person. Jacob is a person. It is usual that a person is reliable. If Jones is reliable then the gunman has a moustache. If Paul is reliable then Jones is not reliable. If Jacob is reliable then Jones is reliable.*
>
> *Paraphrase 2: Jones is a person. Paul is a person. Jacob is a person. If X is a person then it is usual that X is reliable. If Jones is reliable then the gunman has a moustache. If Paul is reliable then Jones is not reliable. If Jacob is reliable then Jones is reliable.*

Our paraphrase 1 takes into consideration some of our caveats above about the introduction and translation of "usually", translating it instead as "it is usual that P". Note the obvious difference between the source and paraphrase 1. The paraphrase is intended to capture the core reasoning in the example, which has as conclusion whether or not the gunman has a moustache. This conclusion depends on the testimony of Jones; this is represented in terms of Jones' reliability. In turn, Jones' reliability is contingent on the testimony of Paul and Jacob, who may or may not be reliable. The rationale for such a paraphrase in our context is that AceRules, in its current form, cannot reason about the contents of subordinate clauses, i.e. the phrase "Jones was looking the other way and did not see what happened", which follows "Paul says that". Moreover, AceRules cannot process the complexity of predicates such as "looking the other way" and "did not see what happened"; similarly for the witness statement of Jacob. The paraphrase has introduced statements about the reliability with respect to 'person', which then requires individuals "Jones", "Paul", and "Jacob" to be asserted as of this class. Finally, considering paraphrase 2, we note that the expression of defeasibility has been represented as a rule about persons "If X is a person then it is usual that X is reliable." The conditional with "it is usual" in the consequence is translated into defeasible rule. As noted above, we model the syntactic form of our natural language treatment of defeasibility "it is usual that" on the form "it is possible that", though with a different form of semantic representation. This discussion highlights that what appear to be relatively

simple statements, e.g. the source above, have several complex elements that need systematic treatment. Other standard examples from the argumentation literature, e.g. the Nixon, Tweety, and Tandem Bicycle examples, also raise issues about representation in AceRules but can nevertheless be treated by our approach.

5 Discussion and Future Work

The paper has outlined an approach to defeasible reasoning in a human-readable way using a CNL, which we justified as a way to *engineer* the range of issues about natural language that must be addressed to facilitate defeasible reasoning. We sketched a motivating example as well as a defeasible inference engine. We articulated the justification for CNLs, outlined AceRules, then highlighted several issues about argumentation in natural language that a CNL must address in future work. Nonetheless, the discussion serves to make the point that simply mining arguments from source text will encounter at least such related problems to instantiate knowledge bases for defeasible reasoning.

While this paper makes foundational progress in argument processing using a CNL, stronger justification for the approach will require working with more substantial and complex knowledge bases. Thus, a major area of future work is to advance the overall pipeline from natural language, formal representation, argument semantics, to verbalised explanations in natural language; such an advance would overcome the limitations of [28]. This will require a systematic, correlated, and iterated development of each component of the processing pipeline. This requires further examples, testing of outputs, identification of translation issues, likely extension of the expressivity of the CNL or argumentation semantics. Relatedly, an important step is to draw upon data derived from argument mining and working to normalising the data into forms suitable to AceRules/dARe. We anticipate that such a step will encounter familiar issues, e.g. pragmatics, ellipsis, and linguistic variation, that arise in formalising natural language.

References

1. Amgoud, L., Besnard, P.: A formal characterization of the outcomes of rule-based argumentation systems. In: SUM. LNCS, vol. 8078, pp. 78–91. Springer (2013)
2. Amgoud, L., Nouioua, F.: Undercutting in argumentation systems. In: SUM. LNCS, vol. 9310, pp. 267–281. Springer (2015)
3. Androutsopoulos, I., Malakasiotis, P.: A survey of paraphrasing and textual entailment methods. Journal of Artificial Intelligence Research 38, 135–187 (2010)
4. Asher, N.: Reference to Abstract Objects in Discourse. Kluwer Academic Publishers (1993)
5. Baral, C., Gelfond, M.: Logic programming and knowledge representation. Journal of Logic Programing 19/20, 73–148 (1994)
6. Bondarenko, A., Dung, P.M., Kowalski, R.A., Toni, F.: An abstract, argumentation-theoretic approach to default reasoning. Artificial Intelligence 93, 63–101 (1997)
7. Caminada, M., Amgoud, L.: On the evaluation of argumentation formalisms. Artificial Intelligence 171(5–6), 286–310 (2007)
8. Craven, R., Toni, F.: Argument graphs and assumption-based argumentation. Artificial Intelligence 233, 1–59 (2016)

9. Dung, P.M.: On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games. Artificial Intelligence 77(2), 321–358 (1995)
10. Dung, P.M., Son, T.C.: An argument-based approach to reasoning with specificity. Artificial Intelligence 133(1-2), 35–85 (2001)
11. Fuchs, N.E.: Reasoning in Attempto Controlled English: Non-monotonicity. In: Davis, B., Pace, G.J., Wyner, A. (eds.) Controlled Natural Language – 5th International Workshop, CNL 2016, Aberdeen, UK, July 25-27, 2016, Proceedings. Lecture Notes in Computer Science, vol. 9767, pp. 13–24. Springer (2016)
12. Fuchs, N.E., Kaljurand, K., Kuhn, T.: Attempto Controlled English for knowledge representation. In: Reasoning Web. pp. 104–124 (2008)
13. Guy, S., Schwitter, R.: The PENG^{ASP} system: Architecture, language and authoring tool. Language Resources and Evaluation pp. 1–26 (2016)
14. Kamp, H., Reyle, U.: From Discourse to Logic: Introduction to Model-theoretic Semantics of Natural Language: Formal Logic and Discourse Representation Theory. Springer (1993)
15. Kratzer, A.: Modals and Conditionals. Oxford University Press (2012)
16. Kuhn, T.: AceRules: Executing rules in controlled natural language. In: Web Reasoning and Rule Systems, First International Conference, RR 2007, Innsbruck , Austria, June 7-8, 2007, Proceedings. Lecture Notes in Computer Science, vol. 4524, pp. 299–308. Springer (2007)
17. Kuhn, T.: A survey and classification of controlled natural languages. Computational Linguistics 40(1), 121–170 (2014)
18. Lippi, M., Torroni, P.: Argumentation mining: State of the art and emerging trends. ACM Transactions on Internet Technology 16(2), 10:1–10:25 (2016)
19. Mott, D.: The ITA controlled english report (Prolog version). Tech. rep., Emerging Technology Services, Hursley, IBM UK (2016)
20. Pollock, J.L.: Reasoning and probability. Law, Probability and Risk 6, 43–58 (2007)
21. Prakken, H.: An abstract framework for argumentation with structured arguments. Argument & Computation 1(2), 93–124 (2010)
22. Silver, D., Huang, A., Maddison, C.J., Guez, A., Sifre, L., van den Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M., Dieleman, S., Grewe, D., Nham, J., Kalchbrenner, N., Sutskever, I., Lillicrap, T., Leach, M., Kavukcuoglu, K., Graepel, T., Hassabis, D.: Mastering the game of Go with deep neural networks and tree search. Nature 529, 484–489 (2016)
23. Strass, H.: Instantiating rule-based defeasible theories in abstract dialectical frameworks and beyond. Journal of Logic and Computation (Feb 2015), advance access `http://dx.doi.org/10.1093/logcom/exv004`
24. Strass, H., Wyner, A.: On automated defeasible reasoning with controlled natural language and argumentation. In: AAAI-17 Workshop on knowledge-based Techniques for Problem Solving and Reasoning (KnowProS 2017) (Feb 2017)
25. Toniolo, A., Preece, A.D., Webberley, W., Norman, T.J., Sullivan, P., Dropps, T.: Conversational intelligence analysis. In: Proceedings of the 17th International Conference on Distributed Computing and Networking, Singapore, January 4-7, 2016. pp. 42:1–42:6 (2016)
26. Wyner, A., Bench-Capon, T., Dunne, P., Cerutti, F.: Senses of 'argument' in instantiated argumentation frameworks. Argument & Computation 6(1), 50–72 (2015)
27. Wyner, A., van Engers, T., Bahreini, K.: From policy-making statements to first-order logic. In: EGOVIS. pp. 47–61 (2010)
28. Wyner, A.Z., van Engers, T.M., Hunter, A.: Working on the argument pipeline: Through flow issues between natural language argument, instantiated arguments, and argumentation frameworks. Argument & Computation 7(1), 69–89 (2016)