Controlled Natural Languages for Knowledge Representation and Reasoning[*]

Tiantian Gao[1]

1 **Department of Computer Science, Stony Brook University**
Stony Brook, NY, USA
`tiagao@cs.stonybrook.edu`

Abstract

Controlled natural languages (CNLs) are effective languages for knowledge representation and reasoning. They are designed based on certain natural languages with restricted lexicon and grammar. CNLs are unambiguous and simple as opposed to their base languages. They preserve the expressiveness and coherence of natural languages. In this paper, it mainly focuses on a class of CNLs, called machine-oriented CNLs, which have well-defined semantics that can be deterministically translated into formal languages to do logical reasoning. Although a number of machine-oriented CNLs emerged and have been used in many application domains for problem solving and question answering, there are still many limitations: First, CNLs cannot handle inconsistencies in the knowledge base. Second, CNLs are not powerful enough to identify different variations of a sentence and therefore might not return the expected inference results. Third, CNLs do not have a good mechanism for defeasible reasoning. This paper addresses these three problems and proposes a research plan for solving these problems. It also shows the current state of research: a paraconsistent logical framework from which six principles that guide the user to encode CNL sentences were created. Experiment results show this paraconsistent logical framework and these six principles can consistently and effectively solve word puzzles with injections of inconsistencies.

1998 ACM Subject Classification I.2.1 Applications and Expert Systems

Keywords and phrases Controlled Natural Languages, Paraconsistent Logics, Defeasible Reasoning

Digital Object Identifier 10.4230/OASIcs.ICLP.2016.19

1 Introduction

Controlled natural languages (CNLs) are effective languages for knowledge representation and reasoning. According to [27], “A controlled natural language is a constructed language that is based on a certain natural language, being more restrictive concerning lexicon, syntax, and/or semantics while preserving most of its natural properties”. Unlike the languages that develop naturally, constructed languages are the languages whose lexicon and syntax are designed with intent. A CNL is constructed on the basis of an existing natural language, such as English, French, or German. Words in the lexicon of a CNL mainly come from its base language. They may or may not be used in the same manner as in the base language. Some words are used with fewer senses or reserved as key-words for specific purposes. CNLs have a well-defined syntax to form phrases, sentences and texts. The syntax of a CNL is generally simpler than that of the source language. Sentences are interpreted in a deterministic way.

[*] This work was partially supported by Michael Kifer and Paul Fodor.

CNLs are more accurate than natural languages, because the language is more restrictive, but not all CNLs have formal semantics. Those that have formal semantics can be processed by computers for knowledge representation, machine translation, and logical reasoning. Although a CNL may deviate from its base language in the lexicon, syntax, and/or semantics, it still preserves most of the natural properties of the base language, so the reader would correctly comprehend the CNL with little effort.

CNLs generally fall into two categories: human-oriented CNLs and machine-oriented CNLs. Human-oriented CNLs are designed to make the texts easier for readers to understand [40]. Examples include Basic English [36], Special English [35], and Simplified Technical English [23]. Machine-oriented CNLs, as opposed to human-oriented ones, have formal semantics which can be understood and processed by computers for the purpose of knowledge representation and logical reasoning. Examples include Attempto Controlled English (ACE) [17], Processable English (PENG) [39], and Computer-processable Language (CPL) [14].

Typically, there is a big learning curve for domain experts in the fields such as law, business and medical to represent the domain-specific knowledge in computer languages. CNLs are superior to other ways of knowledge representation in that they require little knowledge for users to understand the syntax and semantics of the underlying knowledge representation framework. Users can encode the knowledge base in English by following a restricted grammar and then make inferences. For example, in the ERGO project (Effective Representation of Guidelines with Ontologies)[1], ACE was used to author pediatric guideline recommendations. As a result, the clinical practice guidelines can be automatically translated into rules which can be incorporated into decision support systems to facilitate clinicians. In [41], PENG was used to solve word puzzles. The original Jobs Puzzle was rewritten in CNL sentences with the addition of some implicit background knowledge. These CNL sentences are then transformed into a program in Answer Set Programming (ASP) [22] paradigm to compute the answer. In [13], CPL was used to encode the AP (advanced high-school) level examination questions. The CNLs questions are machine-understandable such that they can be processed by its inference system for question-answering.

There are limitations to the aforementioned CNLs. First, they cannot conduct reasoning in the presence of inconsistencies. In practical cases, it is very likely that the knowledge base is constructed from different sources, thus the occurrences of inconsistencies are quite likely. However, current reasoning systems for the aforementioned CNLs do not accept inconsistencies. Because of this, occurrences of inconsistencies in one source will break the whole system and inhibit reasoning. But, in many cases, inconsistencies in one source may not affect the others. Thus, it is necessary to know which piece of information is inconsistent. Besides, it is also desirable to derive things from the information which is consistent.

Second, current CNLs have limited power to identify variations of a sentence. Although CNLs have restricted grammar and pre-defined interpretation rules, users still have multiple choices to express a sentence. Consider the case of question-answering, users may also compose questions in different words as opposed the ones used in the knowledge base or have different ways of writing the same sentence in CNL. In the aspect of knowledge representation, it is desirable to map the variations of a sentence to the same logical form. Otherwise, users might not be able to get the inference results as expected. For instance, in ACE, phrases like "Mary's father" and "the father of Mary" are represented as the same form. However, given the sentences "Mary's gender is female. If Mary is a female then Mary is a doctor", ACE will not derive the conclusion that "Mary is a doctor".

[1] `http://gem.med.yale.edu/ergo/`

Third, it is common that CNL sentences are not created equally. Different sentences imply different degrees of priorities. Consider the sentences "Every bird flies. Penguins do not fly". The first sentence states the default case: a bird flies. The second sentence indicates a higher priority than the first one - in that if something is a penguin, it will refute the conclusion drawn from the first one. This type of reasoning is called defeasible reasoning [34]. Defeasible statements are common in texts. Although reference [42] provides a mechanism to denote defaults and exceptions in CNL, it is still very limited when handling priorities in complex cases. Details will be discussed in the next section.

In the following: Section 2 gives an overview of existing CNLs. Section 3 presents the goal of the research. Section 4 shows the current state of research, to be specific, a powerful paraconsistent logical framework and six principles derived the logic for encoding CNL sentences. Section 5 gives a brief summary of the experiment results show the aforementioned logical framework and principles can consistently and effectively solve word puzzles. Section 6 discusses the open issues and the expected achievements in the future. Section 7 concludes the paper.

2 Background

ACE is the first CNL that can be translated to first-order logic. ACE is a subset of English defined by a restricted grammar along with interpretation rules that control the semantic analysis of grammatically correct ACE sentences. ACE uses discourse resolution structure (DRS) [25] as the logical structure to represent the semantics of a set of ACE sentences. ACE is supported by a language processor, Attempto Parsing Engine (APE), and a reasoner, RACE [19]. APE is an online language processor that allows users to compose ACE sentences as input and generates their semantics in DRS and first-order logic clauses as output. RACE is a CNL reasoner that supports theorem proving, consistency checking, and question answering. RACE is implemented in Prolog. It is an extension of Satchmo [30], which is a theorem prover based on the model generation paradigm. Satchmo executes the clauses by forward reasoning and generates a minimal finite model of clauses. RACE extends Satchmo by giving a justification for every proof, finding all minimal unsatisfiable subsets of clauses if the axioms are not consistent,

PENG was developed by Rolf Schwitter at Macquarie University. It was partly inspired by ACE. PENG is a subset of English with restricted grammar and use DRS as semantic representation. Unlike ACE, PENG does not require users to learn the grammar of the language. Instead, it designs a predictive editor that informs users of the look-ahead information that guides users to proceed based on the structure of the current sentence. The original implementation of PENG's reasoner is based on a theorem-prover Otter [32] and a model builder MACE [33]. The reasoner supports consistency checking, informativity checking, and question answering. In later extensions, PENG translates CNL sentences into ASP programs and embeds Clingo [21] as its underlying reasoner for question answering. Besides, it extends the grammar to support defeasible reasoning [42] by introducing defaults and exceptions. A default statement is identified by the keyword *normally*. There are two types exceptions: strong exception and weak exception, where strong exceptions, identified by the word "not", can refute the default conclusion and weak exceptions, identified by the keyword "abnormally" make the default conclusion inapplicable without refuting it. There are a few limitations to this approach. First, the way it represents weak exceptions is more close to the English translation of the intended ASP rule. It is very hard for users to correctly represent weak exceptions in CNL without knowing the underlying ASP rules. Second, the design of defaults and exceptions only generates two levels of priorities where exceptions have higher priorities than defaults and therefore refuting the defaults. However, in real cases, it is very common, especially in the fields of law and financial regulations, that there are more than two levels of priorities among sentences. A sentence can refute some sentences while the sentence itself can be refuted by others as well.

CPL was developed by Peter Clark at University of Texas. The vocabulary of CPL is based on a pre-defined Component Library (CLib) ontology [8]. CPL accepts three types of sentences: ground facts, rules, and questions. The semantics of CPL are represented by KM (Knowledge Machine) [15] sentences. KM is a powerful frame-based knowledge representation language. It represents first-order logic clauses in LISP-like syntax. The CPL interpreter translates a CPL sentence into KM sentences in three steps. First, the interpreter uses a bottom-up, broad coverage chart parser, called SAPIR [24], to parse a CPL sentence and then generates a logical form (LF). Second, an initial logic generator is used to transform the LF into ground logical assertions (KM sentences) by applying a set of simple, syntactic rewrite rules. Third, subsequent post-processing is performed based on the logical assertions generated in Step 2, including word sense disambiguation, semantic role labelling, and structural re-organization.

BioQuery-CNL [16] is a CNL designed for representing biomedical documents. The expressive power of BioQuery-CNL is superior to existing semantic web query languages, like SPARQL. For instance, users can write simple English phrases such as "gene-gene relation chain" to indicate transitive closures. Both the biomedical knowledge base and user queries are encoded by ASP programs, which can be fed into ASP solvers for making inferences. Between the querying interface and the underlying knowledge base, there is an intermediate layer, the rule layer, which stores definitions of auxiliary concepts derived from the knowledge base. These auxiliary definitions help connect ASP queries to the underlying knowledge base.

NL2KR [43] is a platform that can translate natural languages into knowledge representation formalisms. It consists of two sub-parts: NL2KR-L and NL2KR-T, where NL2KR-L is the training phase of the system and NL2KR-T is the translation system. Both parts embed a Combinatory Categorial Grammar (CCG) [28] parser, where each word is associated with a syntactic category and semantic representation in the form of λ-expressions. The purpose of NL2KR-L is to learn the semantic meaning of each word in the lexicon based on a training set. Given the sentences and their semantic representations, Inverse-λ [6] is used to extract the semantic meaning of a word within the given context. When Inverse-λ is not enough to extract the meaning of the words, Generalization [7] is used to guess the meaning of the words. Ambiguity is solved by a Parameter Learning module which learns the weights of all possible meanings to a word and chooses the most probable one. After the training phase, words in the lexicon are augmented with new meanings extracted from the training set. In the translation phase, sentences are translated by a CCG parser. Same as NL2KR-L, Generalization is used to determine the meaning of unknown words. Experimental results show that NL2KR achieves high accuracy when applied to GeoQuery and Jobs datasets for question-answering.

In addition to CNL systems, current advances in ASP provide ways to solve more complicated knowledge representation problems in CNL. For instance, CR-Prolog [5] extends ASP with consistency-restoring rules (cr-rules), which can be used to specify exceptions. Once inconsistencies arise in the knowledge base, cr-rules are used to override the conclusions derived from default statements. This logical framework captures the characteristics of defeasible reasoning in natural languages. Another extension of ASP is EZCSP [2], which is designed to encode numerical information and reason about it efficiently. This feature can be applied to represent numerical information in natural languages and achieve high performance in reasoning. In addition, there is ASP{f} [3, 4], which augments EZCSP and can handle defaults and exceptions in ASP as well.

3 Goal of the Research

The first goal is to develop a paraconsistent logic that handles inconsistencies in CNL reasoning. Although there is a list of paraconsistent logics, e.g., [37], [10], and [9], they deal with inconsistencies from the philosophical or mathematical point of view. Other paraconsistent logics, such as [11] and [26], were developed for definite logic programs and cannot be easily applied to solving more complex CNL reasoning problems.

There is a list of desired properties the intended paraconsistent logical framework is supposed to have: First, the logic intends to identify the most likely cause of inconsistencies. Consider the knowledge base consisting of the following sentences: 1) Every actor is male, 2) Mary is a female, and 3) Mary is an actor. Apparently, the knowledge base is not consistent since Mary is a female but she is also an actor. There could be two explanations: one where Mary is not an actor and the other where Mary is not a female. Given that Mary is a female name, the former explanation is more reasonable than the latter one. To achieve this goal, it is required that the logic can select the most preferred models by taking into account some background knowledge. Second, in some cases, contrapositive inference is used in CNL reasoning but this is not always the case. Therefore, the logic intends to provide a mechanism to allow/inhibit contrapositive inference. Third, since if + *premise* + then + *conclusion* statements are used to derive new facts, it is necessary that the logical framework has a mechanism to decide whether or not to derive conclusions from inconsistent premises. Last, as closed world assumption [29] is used in databases, this is also useful for CNL reasoning. Therefore, the underlying logic should be able to ensure complete knowledge of information.

The second goal is to standardize logical representations of CNLs, such that they have more power to identify different variations of the same sentence and map them to the same logical form. As is discussed in the introduction, although simple forms of paraphrases of sentences can be identified by CNLs, they are still very limited. For instance, the sentence "Mary has a dog" and "Mary owns a dog" will be translated into two different logical forms in ACE. As a result, users may not get the expected answers when they compose question in a way that uses different terms as in the knowledge base. First, a list of standardized relations is required to be defined to achieve this goal. Second, methods should be proposed to extract the relations from CNL sentences by consulting their syntactic or semantic properties. Although there is a list of tools such as StanfordIE and Ollie for relation extractions, the number of pre-defined relations are very small. Although the structures of CNL sentences are more restricted as opposed to the ones StanfordIE [1] and Ollie [31] work on, the standardization intends to normalize all possible relations in logical representations instead of a few pre-defined relations such as *location*, *founded_by*, etc.

The third goal is to enable defeasible reasoning in CNLs. The previous section shows the limitations of [42] in handling defaults. That is, there can be only two levels of priorities among CNL sentences. To allow more than two levels of priorities for CNL sentences, to the best my knowledge, Logic Programming with Defaults and Argumentation Theories (LPDA) [44] can be considered as a good candidate framework with desirable features.

LPDA is based on the three-valued well-founded semantics [38]. It is a unifying defeasible reasoning framework that uses defaults and exceptions with prioritized rules, and argumentation theories. LPDA has two types of rules: strict and defeasible, where strict rules generate non-defeasible conclusions and defeasible rules generate defeasible conclusions that can be defeated by some exceptions. Each LPAD program is accompanied by an argumentation theory that specifies when a defeasible rule is defeated. A rule is defeated if it is refuted, rebutted, or disqualified. Generally, a rule is refuted if there is another rule that draws an incompatible conclusion with higher priority. A rule is rebutted if there is another rule that draws an incompatible conclusion and there is no way to resolve the conflict based on the relative priorities. A rule is disqualified if it is cancelled, self-defeated, etc. Based on LPDA, defeasible statements in CNL can be encoded by defeasible rules and their priorities can be specified in argumentation theories.

Another challenge is identification of the priorities among CNL sentences. This can be done either explicitly by user specifications or implicitly detected by some background knowledge or natural language understanding methodologies.

4 Current State of Research

Reference [20] shows the current state of research. A new kind of paraconsistent logic was developed to deal with inconsistencies in word puzzles, more generally, for translating CNL sentences into logic. The logical framework is based on the well-known type of paraconsistent logics, Annotated Predicate Calculus (APC) [26], but has a new kind of non-monotonic semantics, called *consistency preferred stable models.* The language is a logic programming subset of APC, denoted as APC_{LP}. APC_{LP} can be isomorphically embedded in ASP extended with a model preference framework, such as the Clingo [21] with its Asprin extension [12]. It was proved in [20] that this embedding is one-to-one and preserves the semantics.

Along with the logical framework, six principles were proposed to guide users to encode CNL sentences in APC_{LP}. Each of the principles will be briefly described in the following: Principle 1 guides users to encode an if + *premise* + then + *conclusion* sentence that can perform contrapositive inference. Principle 2 describes the way to encode if + *premise* + then + *conclusion* sentences such that it can allow/inhibit derivations of conclusions from inconsistent premises. Principle 3 addresses the encoding of polar facts in CNL. For instance, a person must be either a male or a female, but not both or unknown. When inconsistency is possible, this principle ensures this requirement. Besides, if one of them is inconsistent then the other is too. Inconsistent information is not created equal, as people have different degrees of confidence in different pieces of information based on common sense knowledge. For example, there is more confidence in that someone whom people barely know is a person compared to the information about this person's marital situation (e.g., whether a husband exists). Principle 4 allows users to specify the degrees of confidence. As a result, when there are multiple explanations for the cause of inconsistencies, Principle 4 will select the most reasonable one by consulting the degrees specified. Principle 5 behaves like the closed world assumption. It guides users to encode CNL sentences ensure complete knowledge of information. Principle 6 captures the cardinality constraints in the presence of inconsistencies in CNL sentences, e.g., a person holds exactly one job.

5 Preliminary results

The paraconsistent logical framework and the proposed principle mentioned in the previous section have been applied to solve word puzzles, such as Jobs Puzzle [45] and Zebra Puzzle[2] with inconsistencies. Experiment results show that in the cases where there is no inconsistency, APC_{LP} can correctly compute the answer. In the cases of inconsistencies, APC_{LP} can find the most likely cause of inconsistencies within the puzzle and give reasonable inference results. More detailed information can be found in [20].

6 Open Issues and Expected Achievements

The first issue is that current CNLs have limited power to recognize variations of a sentence and therefore might not always map sentences that express the same meaning to the same logical form. As the next step, it is intended to extend ACE to overcome this issue. ACE parser translates CNL sentences into DRS with pre-defined predicates [18] to represent the semantics of a sentence. This form of representation is simple and well-structured. It is intended to do post-processing based on the semantic representation in order to extract semantic relations standardized in ontologies, such as DBpedia[3] and Wikidata[4]. The second issue is to perform defeasible reasoning in CNLs. In order to detect the refutation relations between two sentences, it is expected to do the following: First, extend ACE to incorporate some background knowledge for primitive detection of sentence priorities. Second, design a user interface that allows users to make corrections. In addition, it is expected to extend current DRS representation to accommodate defeasible information, which will be eventually translated to an LPDA program for defeasible reasoning.

7 Conclusion

In this paper, it first gives an overview of the development of CNLs and discusses the limitations of current CNLs in the aspect of knowledge representation and reasoning. Then, it gives an outline of the research plan for solving these problems. This includes designing a paraconsistent logical framework for knowledge representation, empowering current CNLs to recognize variations of a sentence and perform defeasible reasoning. Next, it shows the current state of research - a powerful paraconsistent logical framework along with six principles derived from that for encoding CNL sentences. In addition, it shows the application of the current work to solving word puzzles with inconsistencies. Finally, it addresses some open issues and presents the plans for future achievements.

References

1 Gabor Angeli, Melvin Jose Johnson Premkumar, and Christopher D. Manning. Leveraging linguistic structure for open domain information extraction. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing, ACL 2015, July 26-31, 2015, Beijing, China, Volume 1: Long Papers*,

[2] `https://en.wikipedia.org/wiki/Zebra_Puzzle`
[3] `http://wiki.dbpedia.org/`
[4] `https://www.wikidata.org/wiki/Wikidata:Main_Page`

pages 344–354. Association for Computational Linguistics, The Association for Computer Linguistics, 2015.

2 Marcello Balduccini. Representing constraint satisfaction problems in answer set programming. In *ICLP09 Workshop on Answer Set Programming and Other Computing Paradigms (ASPOCP09)(Jul 2009)*, 2009.

3 Marcello Balduccini. A "conservative" approach to extending answer set programming with non-herbrand functions. In *Correct Reasoning*, pages 24–39. Springer, 2012.

4 Marcello Balduccini. Asp with non-herbrand partial functions: A language and system for practical use. *Theory and Practice of Logic Programming*, 13(4-5):547–561, 2013.

5 Marcello Balduccini and Michael Gelfond. Logic programs with consistency-restoring rules. In *International Symposium on Logical Formalization of Commonsense Reasoning, AAAI 2003 Spring Symposium Series*, volume 102, 2003.

6 Chitta Baral, Juraj Dzifcak, Marcos Alvarez Gonzalez, and Aaron Gottesman. Typed answer set programming lambda calculus theories and correctness of inverse lambda algorithms with respect to them. *TPLP*, 12(4-5):775–791, 2012.

7 Chitta Baral, Juraj Dzifcak, Marcos Alvarez Gonzalez, and Jiayu Zhou. Using inverse lambda and generalization to translate english to formal languages. *CoRR*, abs/1108.3843, 2011.

8 Ken Barker, Bruce W. Porter, and Peter Clark. A library of generic concepts for composing knowledge bases. In *Proceedings of the First International Conference on Knowledge Capture (K-CAP 2001), October 21-23, 2001, Victoria, BC, Canada*, pages 14–21. ACM, 2001.

9 Nuel D Belnap Jr. A useful four-valued logic. In *Modern uses of multiple-valued logic*, pages 5–37. Springer, 1977.

10 Jean-Yves Béziau, Walter Alexandre Carnielli, and Dov M Gabbay. *Handbook of paraconsistency.* College Publications, 2007.

11 Howard A Blair and VS Subrahmanian. Paraconsistent logic programming. In *International Conference on Foundations of Software Technology and Theoretical Computer Science*, pages 340–360. Springer, 1987.

12 Gerhard Brewka, James P. Delgrande, Javier Romero, and Torsten Schaub. asprin: Customizing answer set preferences without a headache. In Blai Bonet and Sven Koenig, editors, *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence, January 25-30, 2015, Austin, Texas, USA.*, pages 1467–1474. AAAI Press, 2015.

13 Peter Clark, Shaw Yi Chaw, Ken Barker, Vinay K. Chaudhri, Philip Harrison, James Fan, Bonnie E. John, Bruce W. Porter, Aaron Spaulding, John A. Thompson, and Peter Z. Yeh. Capturing and answering questions posed to a knowledge-based system. In Derek H. Sleeman and Ken Barker, editors, *Proceedings of the 4th International Conference on Knowledge Capture (K-CAP 2007), October 28-31, 2007, Whistler, BC, Canada*, pages 63–70. ACM, 2007.

14 Peter Clark, Philip Harrison, Thomas Jenkins, John A. Thompson, and Richard H. Wojcik. Acquiring and using world knowledge using a restricted subset of english. In Ingrid Russell and Zdravko Markov, editors, *Proceedings of the Eighteenth International Florida Artificial Intelligence Research Society Conference, Clearwater Beach, Florida, USA*, pages 506–511. AAAI Press, 2005.

15 Peter Clark, Bruce Porter, and Boeing Phantom Works. Km?the knowledge machine 2.0: Users manual. *Department of Computer Science, University of Texas at Austin*, 2:5, 2004.

16 Esra Erdem, Halit Erdogan, and Umut Öztok. BIOQUERY-ASP: querying biomedical ontologies using answer set programming. In Stefano Bragaglia, Carlos Viegas Damásio, Marco Montali, Alun D. Preece, Charles J. Petrie, Mark Proctor, and Umberto Straccia, editors, *Proceedings of the 5th International RuleML2011@BRF Challenge, co-located with*

the 5th International Rule Symposium, Fort Lauderdale, Florida, USA, November 3-5, 2011, volume 799 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2011.

17 Norbert E. Fuchs, Kaarel Kaljurand, and Tobias Kuhn. Attempto controlled english for knowledge representation. In Cristina Baroglio, Piero A. Bonatti, Jan Maluszynski, Massimo Marchiori, Axel Polleres, and Sebastian Schaffert, editors, *Reasoning Web, 4th International Summer School 2008, Venice, Italy, September 7-11, 2008, Tutorial Lectures*, volume 5224 of *Lecture Notes in Computer Science*, pages 104–124. Springer, 2008.

18 Norbert E. Fuchs, Kaarel Kaljurand, and Tobias Kuhn. Discourse Representation Structures for ACE 6.6. Technical Report ifi-2010.0010, Department of Informatics, University of Zurich, Zurich, Switzerland, 2010.

19 Norbert E. Fuchs and Uta Schwertel. Reasoning in attempto controlled english. In François Bry, Nicola Henze, and Jan Maluszynski, editors, *Principles and Practice of Semantic Web Reasoning, International Workshop, PPSWR 2003, Mumbai, India, December 8, 2003, Proceedings*, volume 2901 of *Lecture Notes in Computer Science*, pages 174–188. Springer, 2003.

20 Tiantian Gao, Paul Fodor, and Michael Kifer. Paraconsistency and word puzzles. *CoRR*, abs/1608.01338, 2016.

21 Martin Gebser, Benjamin Kaufmann, Roland Kaminski, Max Ostrowski, Torsten Schaub, and Marius Thomas Schneider. Potassco: The potsdam answer set solving collection. *AI Commun.*, 24(2):107–124, 2011.

22 Michael Gelfond and Yulia Kahl. *Knowledge representation, reasoning, and the design of intelligent agents: The answer-set programming approach.* Cambridge University Press, 2014.

23 ASD Simplified Technical English Maintenance Group. *ASD-STE 100: Simplified Technical English : International Specification for the Preparation of Maintenance Documentation in a Controlled Language.* Aerospace and Defence Industries Association of Europe, 2007.

24 Philip Harrison and Michael Maxwell. A new implementation of gpsg. In *Proc. 6th Canadian Conf on AI*, pages 78–83, 1986.

25 Hans Kamp and Uwe Reyle. *From discourse to logic: Introduction to modeltheoretic semantics of natural language, formal logic and discourse representation theory*, volume 42. Springer Science & Business Media, 2013.

26 Michael Kifer and Eliezer L. Lozinskii. A logic for reasoning with inconsistency. *J. Autom. Reasoning*, 9(2):179–215, 1992.

27 Tobias Kuhn. A survey and classification of controlled natural languages. *Computational Linguistics*, 40(1):121–170, 2014.

28 Tom Kwiatkowski, Luke S. Zettlemoyer, Sharon Goldwater, and Mark Steedman. Inducing probabilistic CCG grammars from logical form with higher-order unification. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing, EMNLP 2010, 9-11 October 2010, MIT Stata Center, Massachusetts, USA, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1223–1233. ACL, 2010.

29 Vladimir Lifschitz. Closed-world databases and circumscription. *Artif. Intell.*, 27(2):229–235, 1985.

30 Rainer Manthey and François Bry. Satchmo: a theorem prover implemented in prolog. In *International Conference on Automated Deduction*, pages 415–434. Springer, 1988.

31 Mausam, Michael Schmitz, Stephen Soderland, Robert Bart, and Oren Etzioni. Open language learning for information extraction. In Jun'ichi Tsujii, James Henderson, and Marius Pasca, editors, *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, EMNLP-CoNLL 2012, July 12-14, 2012, Jeju Island, Korea*, pages 523–534. ACL, 2012.

32 William McCune. *Otter 3.0 reference manual and guide*, volume 9700. Argonne National Laboratory Argonne, IL, 1994.

33 William McCune. Mace4 reference manual and guide. *arXiv preprint cs/0310055*, 2003.

34 Donald Nute. Defeasible logic, handbook of logic in artificial intelligence and logic programming (vol. 3): nonmonotonic reasoning and uncertain reasoning, 1994.

35 Voice of America (Organization). *VOA Special English word book: a list of words used in Special English programs on radio, television, and the Internet.* Voice of America, 2007.

36 Charles Kay Ogden. *Basic English: A general introduction with rules and grammar.* Number 29 in Psyche miniatures., General series. K. Paul, Trench, Trubner, 1944.

37 Graham Priest, Koji Tanaka, and Zach Weber. *Paraconsistent logic.* Ḿunchen, 1989.

38 Teodor C. Przymusinski. Well-founded and stationary models of logic programs. *Ann. Math. Artif. Intell.*, 12(3-4):141–187, 1994.

39 Rolf Schwitter. English as a formal specification language. In *13th International Workshop on Database and Expert Systems Applications (DEXA 2002), 2-6 September 2002, Aix-en-Provence, France*, pages 228–232. IEEE Computer Society, 2002.

40 Rolf Schwitter. Controlled natural languages for knowledge representation. In Chu-Ren Huang and Dan Jurafsky, editors, *COLING 2010, 23rd International Conference on Computational Linguistics, Posters Volume, 23-27 August 2010, Beijing, China*, pages 1113–1121. Chinese Information Processing Society of China, 2010.

41 Rolf Schwitter. The jobs puzzle: Taking on the challenge via controlled natural language processing. *TPLP*, 13(4-5):487–501, 2013.

42 Rolf Schwitter. Working with defaults in a controlled natural language. In *Australasian Language Technology Association Workshop 2013*, page 106, 2013.

43 Nguyen H Vo, Arindam Mitra, and Chitta Baral. The nl2kr platform for building natural language translation systems. In *Association for Computational Linguistics (ACL)*, 2015.

44 Hui Wan, Benjamin N. Grosof, Michael Kifer, Paul Fodor, and Senlin Liang. Logic programming with defaults and argumentation theories. In Patricia M. Hill and David Scott Warren, editors, *Logic Programming, 25th International Conference, ICLP 2009, Pasadena, CA, USA, July 14-17, 2009. Proceedings*, volume 5649 of *Lecture Notes in Computer Science*, pages 432–448. Springer, 2009.

45 L. Wos. *Automated reasoning: introduction and applications.* McGraw-Hill, 1992.