Solving Logic Puzzles: From Robust Processing to Precise Semantics

Iddo Lev,* Bill MacCartney,* Christopher D. Manning,*† and Roger Levy†

* Department of Computer Science
Stanford University
Stanford, CA 94305-9040, USA
{*iddolev*|*wcmac*|*manning*}*@cs.stanford.edu*

† Department of Linguistics
Stanford University
Stanford, CA 94305-2150, USA
rog@stanford.edu

Abstract

This paper presents intial work on a system that bridges from robust, broad-coverage natural language processing to precise semantics and automated reasoning, focusing on solving logic puzzles drawn from sources such as the Law School Admission Test (LSAT) and the analytic section of the Graduate Record Exam (GRE). We highlight key challenges, and discuss the representations and performance of the prototype system.

1 Introduction

Traditional approaches to natural language understanding (Woods, 1973; Warren and Pereira, 1982; Alshawi, 1992) provided a good account of mapping from surface forms to semantic representations, when confined to a very limited vocabulary, syntax, and world model, and resulting low levels of syntactic/semantic ambiguity. It is, however, difficult to scale these methods to unrestricted, general-domain natural language input because of the overwhelming problems of grammar coverage, unknown words, unresolvable ambiguities, and incomplete domain knowledge. Recent work in NLP has consequently focused on more robust, broad-coverage techniques, but with the effect of overall shallower levels of processing. Thus, state-of-the-art work on probabilistic parsing (e.g., (Collins, 1999)) provides a good solution to robust, broad coverage parsing with automatic and frequently successful ambiguity resolution, but has largely ignored issues of semantic interpretation. The field of Question Answering (Pasca and Harabagiu, 2001; Moldovan et al., 2003) focuses on simple-fact queries. And so-called semantic parsing (Gildea and Jurafsky, 2002) provides as end output only a flat classification of semantic arguments of predicates, ignoring much of the semantic content, such as quantifiers.

A major research question that remains unanswered is whether there are methods for getting from a robust "parse-anything" statistical parser to a semantic representation precise enough for knowledge representation and automated reasoning, without falling afoul of the same problems that stymied the broad application of traditional approaches. This paper presents initial work on a system that addresses this question. The chosen task is solving *logic puzzles* of the sort found in the Law School Admission Test (LSAT) and the old analytic section of the Graduate Record Exam (GRE) (see Figure 1 for a typical example). The system integrates statistical parsing, "on-the-fly" combinatorial synthesis of semantic forms, scope- and reference-resolution, and precise semantic representations that support the inference required for solving the puzzles. Our work complements research in semantic parsing and TREC-style Question Answering by emphasizing complex yet robust inference over general-domain NL texts given relatively minimal lexical and knowledge-base resources.

1.1 Why Logic Puzzles?

Logic puzzles have a number of attractive characteristics as a target domain for research placing a premium on precise inference.

First, whereas for humans the language understanding part of logic puzzles is trivial but the reasoning is difficult, for computers it is clearly the reverse. It is straightforward for a computer to solve a formalized puzzle, so the research effort is on the NLP parts rather than a difficult back-end AI problem. Moreover, only a small core of world knowledge (prominently, temporal and spatial entailments) is typically crucial to solving the task.

Second, the texts employ *everyday language*: there are no domain-restrictions on syntactic and semantic constructions, and the situations described by the texts are diverse.

Third, and most crucial, answers to puzzle questions never explicitly appear in the text and

Preamble: Six sculptures – C, D, E, F, G, and H – are to be exhibited in rooms 1, 2, and 3 of an art gallery. The exhibition conforms to the following conditions:

(1) Sculptures C and E may not be exhibited in the same room.
(2) Sculptures D and G must be exhibited in the same room.
(3) If sculptures E and F are exhibited in the same room, no other sculpture may be exhibited in that room.
(4) At least one sculpture must be exhibited in each room, and no more than three sculptures may be exhibited in any room.

Question 1: If sculpture D is exhibited in room 3 and sculptures E and F are exhibited in room 1, which of the following may be true?

(A) Sculpture C is exhibited in room 1.
(B) No more than 2 sculptures are exhibited in room 3.
(C) Sculptures F and H are exhibited in the same room.
(D) Three sculptures are exhibited in room 2.
(E) Sculpture G is exhibited in room 2.

Question 2: If sculptures C and G are exhibited in room 1, which of the following may NOT be a complete list of the sculpture(s) exhibited in room 2?

(A) Sculpture D (B) Sculptures E and H (C)...

Adapted from (Weber, 1999).

Figure 1: Example of a Puzzle Text

must be *logically inferred* from it, so there is very little opportunity to use existing superficial analysis methods of information-extraction and question-answering as a substitute for deep understanding. A prerequisite for successful inference is *precise* understanding of semantic phenomena like modals and quantifiers, in contrast with much current NLP work that just ignores such items. We believe that representations with a well-defined model-theoretic semantics are required.

Finally, the task has a clear evaluation metric because the puzzle texts are designed to yield exactly one correct answer to each multiple-choice question. Moreover, the domain is another example of "*found* test material" in the sense of (Hirschman et al., 1999): puzzle texts were developed with a goal independent of the evaluation of natural language processing systems, and so provide a more realistic evaluation framework than specially-designed tests such as TREC QA.

While our current system is not a real world application, we believe that the methods being developed could be used in applications such as a computerized office assistant that must understand requests such as: "Put *each* file containing *a* task description in *a different* directory."

2 System Overview

This section explains the languages we use to represent the content of a puzzle. Computing the representations from a text is a complex process with several stages, as shown in Figure 2. Most of the stages are independent of the puzzles domain. Section 3 reviews the main challenges in this process, and later sections outline the various processing stages. More details of some of these stages can be found at (Stanford NLP Group, 2004).

2.1 First-Order Logic (FOL)

An obvious way of solving logic puzzles is to use off-the-shelf FOL *reasoners*, such as theorem provers and model builders. Although most GRE logic puzzles can also be cast as constraint-satisfaction problems (CSPs), FOL representations are more general and more broadly applicable to other domains, and they are closer to the natural language semantics. GRE logic puzzles have finite small domains, so it is practicable to use FOL reasoners.

The ultimate representation of the content of a puzzle is therefore written in FOL. For example, the representation for the first part of constraint (4) in Figure 1 is: $\forall x.room(x) \rightarrow \exists y.sculpture(y) \wedge exhibit(y, x)$. (The treatment of the modal 'must' is explained in §9.2).

2.2 Semantic Logic (SL)

Representing the meaning of natural language texts in FOL is not straightforward because human languages employ events, plural entities, modal operations, and complex numeric expressions. We therefore use an intermediate representation, written in Semantic Logic (SL), which is intended to be a general-purpose semantic representation language. SL extends FOL with event and group variables, the modal operators $\Box$ (necessarily) and $\Diamond$ (possibly), and Generalized Quantifiers (Barwise and Cooper,

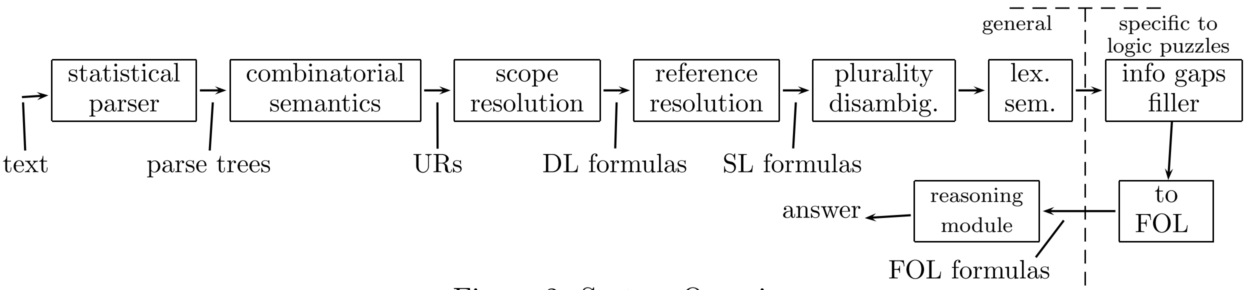

Figure 2: System Overview

1981) $Q(type, var, restrictor, body)$, where *type* can be $\forall$, $\exists$, *at-least*(n), etc. To continue the example, the intermediate representation for the constraint is:

$$\Box Q(\forall, x_1, room(x_1), Q(\geq_1, x_2, sculpture(x_2), \exists e.exhibit(e) \wedge subj(e, x_2) \wedge in(e, x_1)))$$

2.3 Non-determinism

Although logic puzzles are carefully designed to reduce ambiguities to ensure that there is exactly one correct answer per question, there are still many ambiguities in the analysis, such as multiple possibilities for syntactic structures, pronominal reference, and quantifier scope. Each module *ranks* possible output representations; in the event that a later stage reveals an earlier choice to be wrong (it may be inconsistent with the rest of the puzzle, or lead to a non-unique correct answer to a question), the system backtracks and chooses the next-best output representation for the earlier stage.

3 Challenges

3.1 Combinatorial Semantics

The challenge of combinatorial semantics is to be able to assign exactly one semantic representation to each word and sub-phrase regardless of its surrounding context, and to combine these representations in a systematic way until the representation for the entire sentence is obtained. There are many linguistic constructions in the puzzles whose compositional analysis is difficult, such as a large variety of noun-phrase structures (e.g., "Every sculpture must be exhibited in *a different* room") and ellipses (e.g., "Brian saw a taller man than Carl [did]").

3.2 Scope Ambiguities

A sentence has a scope ambiguity when quantifiers and other operators in the sentence can have more than one relative scope. E.g., in constraint (4) of Figure 1, "each room" outscopes "at least one sculpture", but in other contexts, the reverse scoping is possible. The challenge is to find, out of all the possible scopings, the appropriate one, to understand the text as the writer intended.

3.3 Reference Resolution

The puzzle texts contain a wide variety of anaphoric expressions, including pronouns, definite descriptions, and anaphoric adjectives. The challenge is to identify the possible antecedents that these expressions refer to, and to select the correct ones. The problem is complicated by the fact that anaphoric expressions interact with quantifiers and may not refer to any particular context element. E.g., the anaphoric expressions in "Sculptures C and E are exhibited in *the same room*" and in "Each man saw *a different woman*" interact with sets ($\{C, E\}$ and the set of all men, respectively).

3.4 Plurality Disambiguation

Sentences that include plural entities are potentially ambiguous between different readings: distributive, collective, cumulative, and combinations of these. For example, sentence 1 in Figure 1 says (among other things) that *each of* the six sculptures is displayed in *one of* the three rooms – the group of sculptures and the group of rooms behave differently here. Plurality is a thorny topic which interacts in complex ways with other semantic issues, including quantification and reference.

3.5 Lexical Semantics

The meaning of open-category words is often irrelevant to solving a puzzle. For example, the meaning of "exhibited", "sculpture", and "room" can be ignored because it is enough to understand that the first is a binary relation that holds between elements of groups described by the second and third words.[1] This observa-

[1]The meanings are still important for the implicit knowledge that a sculpture cannot be exhibited in more than one room. However, such knowledge can be guessed, as explained in §8.

tion provides the potential for a general system that solves logic puzzles.

Of course, in many cases, the particular meaning of open-category words and other expressions is crucial to the solution. An example is provided in question 2 of Figure 1: the system has to understand what "a complete list" means. Therefore, to finalize the meaning computed for a sentence, such expressions should be expanded to their explicit meaning. Although there are many such cases and their analysis is difficult, we anticipate that it will be possible to develop a relatively compact library of critical puzzle text expressions. We may also be able to use existing resources such as WordNet and FrameNet.

3.6 Information Gaps

Natural language texts invariably assume some knowledge implicitly. E.g., Figure 1 does not explicitly specify that a sculpture may not be exhibited in more than one room at the same time. Humans know this implicit information, but a computer reasoning from texts must be given it explicitly. Filling these information gaps is a serious challenge; representation and acquisition of the necessary background knowledge are very hard AI problems. Fortunately, the puzzles domain allows us to tackle this issue, as explained in §8.

3.7 Presuppositions and Implicatures

In addition to its semantic *meaning*, a natural language text conveys two other kinds of content.

Presuppositions are pieces of information assumed in a sentence. Anaphoric expressions bear presuppositions about the existence of entities in the context; the answer choice "Sculptures C and E" conveys the meaning $\{C, E\}$, but has the presupposition $sculpture(C) \wedge sculpture(E)$; and a question of the form $A \rightarrow B$, such as question 1 in Figure 1, presupposes that A is consistent with the preamble.

Implicatures are pieces of information suggested by the very fact of saying, or not saying, something. Two maxims of (Grice, 1989) dictate that each sentence should be both consistent and informative (i.e. not entailed) with respect to its predecessors. Another maxim dictates saying as much as required, and hence the sentence "No more than three sculptures may be exhibited in any room" carries the implicature that in some possible solution, three sculptures are indeed exhibited in the same room.

Systematic calculation of presuppositions and implicatures has been given less attention in NLP and is less understood than the calculation of meaning. Yet computing and verifying them can provide valuable hints to the system whether it understood the meaning of the text correctly.

4 Morpho-Syntactic Analysis

While traditional hand-built grammars often include a rich semantics, we have found their coverage inadequate for the logic puzzles task. For example, the English Resource Grammar (Copestake and Flickinger, 2000) fails to parse any of the sentences in Figure 1 for lack of coverage of some words and of several different syntactic structures; and parsable simplified versions of the text produce dozens of unranked parse trees. For this reason, we use a broad-coverage statistical parser (Klein and Manning, 2003) trained on the Penn Treebank. In addition to robustness, treebank-trained statistical parsers have the benefit of extensive research on accurate ambiguity resolution. Qualitatively, we have found that the output of the parser on logic puzzles is quite good (see §10). After parsing, each word in the resulting parse trees is converted to base form by a stemmer.

A few tree-transformation rules are applied on the parse trees to make them more convenient for combinatorial semantics. Most of them are general, e.g. imposing a binary branching structure on verb phrases, and grouping expressions like "more than". A few of them correct some parsing errors, such as nouns marked as names and vice-versa. There is growing awareness in the probabilistic parsing literature that mismatches between training and test set genre can degrade parse accuracy, and that small amounts of correct-genre data can be more important than large amounts of wrong-genre data (Gildea, 2001); we have found corroborating evidence in misparsings of noun phrases common in puzzle texts, such as "Sculptures C and E", which do not appear in the Wall Street Journal corpus. Depending on the severity of this problem, we may hand-annotate a small amount of puzzle texts to include in parser training data.

5 Combinatorial Semantics

Work in NLP has shifted from hand-built grammars that need to cover explicitly every sentence structure and that break down on unexpected inputs to more robust statistical parsing.

However, grammars that involve precise semantics are still largely hand-built (e.g. (Carpenter, 1998; Copestake and Flickinger, 2000)). We aim at extending the robustness trend to the semantics. We start with the compositional semantics framework of (Blackburn and Bos, 2000; Bos, 2001) and modify it to achieve greater robustness and coverage.[2]

One difference is that our lexicon is kept very small and includes only a few words with special semantic entries (like pronouns, connectives, and numbers). Open-category words come with their part-of-speech information in the parse trees (e.g. (NN dog)), so their semantics can be obtained using generic semantic templates (but cf. §3.5).

In classic rule-to-rule systems of semantics like (Blackburn and Bos, 2000), each syntactic rule has a separate semantic combination rule, and so the system completely fails on unseen syntactic structures. The main distinguishing goal of our approach is to develop a more robust process that does not need to explicitly specify how to cover every bit of every sentence. The system incorporates a few initial ideas in this direction.

First, role and argument-structure information for verbs is expensive to obtain and unreliable anyway in natural texts. So to deal with verbs and VPs robustly, their semantics in our system exports only an event variable rather than variables for the subject, the direct object, etc. VP modifiers (such as PPs and ADVPs) combine to the VP by being applied on the exported event variable. NP modifiers (including the sentence subject) are combined to the event variable through generic roles: $subj$, np_1, np_2, etc. The resulting generic representations are suitable in the puzzles domain because usually only the relation between objects is important and not their particular roles in the relation. This is true for other tasks as well, including some broad-coverage question answering.

All NPs are analyzed as generalized quantifiers, but a robust compositional analysis for the internal semantics of NPs remains a serious challenge. For example, the NP "three rooms" should be analyzed as $Q(num(3), x, room(x), ..)$, but the word "three" by itself does not contribute the quantifier – compare with "at least three rooms" $Q(\geq_3, x, room(x), ..)$. Yet another case is "the three rooms" (which presupposes a group g such that $g \subseteq room \wedge |g| = 3$). The system currently handles a number of NP structures by scanning the NP left-to-right to identify important elements. This may make it easier than a strictly compositional analysis to extend the coverage to additional cases.

All other cases are handled by a flexible combination process. In case of a single child, its semantics is copied to its parent. With more children, all combinations of applying the semantics of one child to its siblings are tried, until an application does not raise a type error (variables are typed to support type checking). This makes it easier to extend the coverage to new grammatical constructs, because usually only the lexical entry needs to be specified, and the combination process takes care to apply it correctly in the parse tree.

6 Scope Resolution

One way of dealing with scope ambiguities is by using underspecified representations (URs). A UR is a meta-language construct, describing a set of object-language formulas.[3] It describes the pieces shared by these formulas, but possibly underspecifies how they combine with each other. A UR can then be *resolved* to the specific readings it implicitly describes.

We use an extension of Hole Semantics (Blackburn and Bos, 2000)[4] for expressing URs and calculating them from parse trees (modulo the modifications in §5). There are several advantages to this approach. First, it supports the calculation of just one UR per sentence in a combinatorial process that visits each node of the parse tree once. This contrasts with approaches such as Categorial Grammars (Carpenter, 1998), which produce explicitly all the scopings by using type raising rules for different combinations of scope, and require scanning the entire parse tree once per scoping.

Second, the framework supports the expression of scoping constraints between different parts of the final formula. Thus it is possible to express hierarchical relations that must exist between certain quantifiers, avoiding the problems of naive approaches such as Cooper storage (Cooper, 1983). The expression of scoping constraints is not limited to quantifiers and is applicable to all other operators as well. Moreover, it is possible to express scope islands by

[2]Our system uses a reimplementation in Lisp rather than their Prolog code.

[3]In our case, DL formulas – see footnote 6.

[4]The approach is similar to MRS (Copestake et al., 2003).

constraining all the parts of a subformula to be outscoped by a particular node.

Another advantage is that URs support efficient elimination of logically-equivalent readings. Enumerating all scopings and using a theorem-prover to determine logical equivalences requires $O(n^2)$ comparisons for n scopings. Instead, filtering methods (Chaves, 2003) can add tests to the UR-resolution process, disallowing certain combinations of operators. Thus, only one ordering of identical quantifiers is allowed, so "A man saw a woman" yields only one of its two equivalent scopings. We also filter $\forall\Box$ and $\exists\Diamond$ combinations, allowing only the equivalent $\Box\forall$ and $\Diamond\exists$. However, numeric quantifiers are not filtered (the two scopings of "Three boys saw three films" are not equivalent). Such filtering can result in substantial speed-ups for sentences with a few quantifiers (see (Chaves, 2003) for some numbers).

Finally, our true goal is determining the correct relative scoping in context rather than enumerating all possibilities. We are developing a probabilistic scope resolution module that learns from hand-labeled training examples to predict the most probable scoping, using features such as the quantifiers' categories and their positions and grammatical roles in the sentence.[5]

7 Reference Resolution

SL is not convenient for representing directly the meaning of referring expressions because (as in FOL) the extent of a quantifier in a formula cannot be extended easily to span variables in subsequent formulas. We therefore use Discourse Logic (DL), which is SL extended with DRSes and α-expressions as in (Blackburn and Bos, 2000) (which is based on Discourse Representation Theory (Kamp and Reyle, 1993) and its recent extensions for dealing with presuppositions).[6] This approach (like other dynamic semantics approaches) supports the introduction of entities that can later be referred back to, and explains when indefinite NPs should be interpreted as existential or universal quantifiers (such as in the antecedent of conditionals). The reference resolution framework from (Blackburn and Bos, 2000) provides a basis for finding all possible resolutions, but does not specify which one to choose. We are working on a probabilistic reference-resolution module, which will pick from the legal resolutions the most probable one based on features such as: distance, gender, syntactic place and constraints, etc.

8 Filling Information Gaps

To find a unique answer to every question of a puzzle, background information is required beyond the literal meaning of the text. In Question 1 of Figure 1, for example, without the constraint that a sculpture may not be exhibited in multiple rooms, answers B, D and E are all correct. Human readers deduce this implicit constraint from their knowledge that sculptures are physical objects, rooms are locations, and physical objects can have only one location at any given time. In principle, such information could be derived from ontologies. Existing ontologies, however, have limited coverage, so we also plan to leverage information about expected puzzle structures.

Most puzzles we collected are formalizable as constraints on possible tuples of objects. The crucial information includes: (a) the object *classes*; (b) the *constants* naming the objects; and (c) the *relations* used to link objects, together with their arguments' classes. For the sculptures puzzle, this information is: (a) the classes are *sculpture* and *room*; (b) the constants are C, D, E, F, G, H for *sculpture* and $1, 2, 3$ for *room*; (c) the relation is $exhibit(sculpture, room)$. This information is obtainable from the parse trees and SL formulas.

Within this framework, implicit world knowledge can often be recast as mathematical properties of relations. The unique location constraint on sculptures, for example, is equivalent to constraining the mapping from sculptures to rooms to be *injective* (one-to-one); other cases exist of constraining mappings to be *surjective* (onto) and/or *total.* Such properties can be obtained from various sources, including cardinality of object classes, pure lexical semantics, and even through a systematic search for sets of implicit constraints that, in combination with the explicitly stated constraints, yield exactly one answer per question. Figure 3 shows the num-

[5] E.g. there is a strong preference for 'each' to take wide scope, a moderate preference for the first quantifier in a sentence to take wide scope, and a weak preference for a quantifier of the grammatical subject to take wide scope.

[6] Thus, the URs calculated from parse trees are actually URs of DL formulas. The scope resolution phase resolves the URs to explicit DL formulas, and the reference resolution phase converts these formulas to SL formulas.

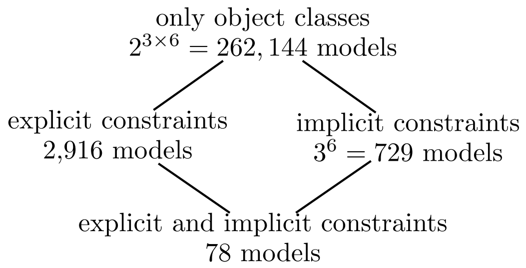

Figure 3: Effect of explicit and implicit constraints on constraining the number of possible models

ber of possible models for the sculptures puzzle as affected by explicit and implicit constraints in the preamble.

9 Solving the Puzzle

9.1 Expanding the answer choices

The body of a logic puzzle question contains a (unique) *wh*-term (typically "which of the following"), a modality (such as "must be true" or "could be true"), and (possibly) an added condition. Each answer choice is expanded by substituting its SL form for the *wh*-term in the question body. For example, the expansion for answer choice (A) of question 1 in Figure 1 would be the SL form corresponding to: "If sculpture D is exhibited . . . , then [*Sculpture C is exhibited in room 1*] must be true".

9.2 Translating SL to FOL

To translate an SL representation to pure FOL, we eliminate event variables by replacing an SL form $\exists e.P(e) \wedge R_1(e, t_1) \wedge .. \wedge R_n(e, t_n)$ with the FOL form $P(t_1, .., t_n)$. An ordering is imposed on role names to guarantee that arguments are always used in the same order in relations. Numeric quantifiers are encoded in FOL in the obvious way, e.g., $Q(\geq_2, x, \varphi, \psi)$ is translated to $\exists x_1 \exists x_2.\ x_1 \neq x_2 \wedge (\varphi \wedge \psi)[x_1/x] \wedge (\varphi \wedge \psi)[x_2/x]$.

Each expanded answer choice contains one modal operator. Modals are moved outward of negation as usual, and outward of conditionals by changing $A \rightarrow \Box B$ to $\Box(A \rightarrow B)$ and $A \rightarrow \Diamond B$ to $\Diamond(A \wedge B)$. A modal operator in the outermost scope can then be interpreted as a directive to the reasoning module to test either entailment ($\Box$) or consistency ($\Diamond$) between the preamble and the expanded answer choice.

9.3 Using FOL reasoners

There are two reasons for using both theorem provers and model builders. First, they are complementary reasoners: while a theorem prover is designed to demonstrate the inconsistency of a set of FOL formulas, and so can find the correct answer to "must be true" questions through proof by contradiction, a model builder is designed to find a satisfying model, and is thus suited to finding the correct answer to "could be true" questions.[7] Second, a reasoner may take a very long time to halt on some queries, but the complementary reasoner may still be used to answer the query in the context of a multiple-choice question through a *process of elimination*. Thus, if the model builder is able to show that the negations of four choices are consistent with the preamble (indicating they are not entailed), then it can be concluded that the remaining choice is entailed by the preamble, even if the theorem prover has not yet found a proof.

We use the Otter 3.3 theorem prover and the MACE 2.2 model builder (McCune, 1998).[8] The reasoning module forks parallel sub-processes, two per answer choice (one for Otter, one for MACE). If a reasoner succeeds for an answer choice, the choice is marked as correct or incorrect, and the dual sub-process is killed. If all answer-choices but one are marked incorrect, the remaining choice is marked correct even if its sub-processes did not yet terminate.

10 Progress

Using the sculptures puzzle (a set of four questions partly shown in Figure 1) as an initial test case, we have built a prototype end-to-end system. In its present state, the system analyzes and solves correctly all questions in this puzzle, except that there is still no understanding of the phrase "complete list" in question 2. The back-end reasoning module is finished and works for any puzzle formalized in FOL+modals. The probabilistic scope resolution module, trained on 259 two-quantifier sentences extracted from 122 puzzles and tested on 46 unseen sentences, attains an accuracy of about 94% over an 82% linear-order baseline. A preliminary evaluation on another unseen puzzle shows that on 60% of the sentences, the parser's output is accurate enough to support the subsequent computation of the semantics, and we expect this to be better after it is trained on puzzle texts. However, the

[7] GRE puzzles always deal with finite domains, so a model builder is guaranteed to halt on consistent sets of formulas.

[8] An advantage of using Otter and MACE is that they are designed to work together, and use the same input syntax.

system as a whole worked end-to-end on only one of the unseen sentences in that puzzle; key losses come from unhandled semantic phenomena (e.g. “only”, “except”, ellipses), unhandled lexical semantics of words that must be understood (e.g. “complete list”), and unhandled implicit constraint types that need to be filled.

11 Conclusion and Further Work

The key open problem is identifying sufficiently robust and general methods for building precise semantic representations, rather than requiring hand-built translation rules for a seemingly endless list of special phenomena. Immediate future work will include extending and generalizing the system's coverage of syntax-to-semantics mappings, incorporating classifiers for suggesting likely coreference resolutions and operator scopings, and developing methods for calculating presuppositions and inferences. This work may be sufficient to give good coverage of the problem domain, or we may need to develop new more robust models of syntactic to semantic transductions.

Acknowledgements

Thanks to Kristina Toutanova for useful discussions.

This work was supported in part by the Advanced Research and Development Activity (ARDA)'s Advanced Question Answering for Intelligence (AQUAINT) Program; in part by the Department of the Navy under grant no. N000140010660, a Multidisciplinary University Research Initiative on Natural Language Interaction with Intelligent Tutoring Systems; and in part by Department of Defense award no. NBCH-D-03-0010(1) “A Person-Centric Enduring, Personalized, Cognitive Assistant.”

References

Hiyan Alshawi, editor. 1992. *The Core Language Engine.* MIT Press.

J. Barwise and R. Cooper. 1981. Generalized quantifiers and natural language. *Linguistics and Philosophy*, 4:159–219.

Patrick Blackburn and Johan Bos. 2000. *Representation and Inference for Natural Language: A First Course in Computational Semantics.* http://www.comsem.org/.

Johan Bos. 2001. Doris 2001: Underspecification, resolution and inference for discourse representation structures. In Blackburn and Kohlhase, editors, *ICoS-3. Inference in Compuational Semantics. Workshop Proceedings.*

Bob Carpenter. 1998. *Type-Logical Semantics.* MIT Press.

Rui P. Chaves. 2003. Non-redundant scope disambiguation in underspecified semantics. In Balder ten Cate, editor, *Proc. of the 8th ESSLLI Student Session*, pages 47–58.

Michael Collins. 1999. *Head-Driven Statistical Models for Natural Language Parsing.* Ph.D. thesis, University of Pennsylvania.

Robin Cooper. 1983. *Quantification and Syntactic Theory.* Reidel, Dordrecht.

A. Copestake and D. Flickinger. 2000. An open-source grammar development environment and broad-coverage english grammar using HPSG. In *Proceedings of LREC.*

A. Copestake, D. Flickinger, C. Pollard, and I. Sag. 2003. Minimal recursion semantics: an introduction. http://lingo.stanford.edu/sag/publications.html.

D. Gildea and D. Jurafsky. 2002. Automatic labeling of semantic roles. *Computational Linguistics*, 28(3):245–288.

Daniel Gildea. 2001. Corpus variation and parser performance. In *Proceedings of EMNLP*, pages 167–202.

Paul H. Grice. 1989. *Studies in the way of words.* Harvard University Press.

Lynette Hirschman, Marc Light, Eric Breck, and John D. Burger. 1999. Deep Read: A reading comprehension system. In *Proc. of the 37th Annual Meeting of the ACL*, pages 325–332.

Hans Kamp and Uwe Reyle. 1993. *From Discourse to Logic.* Kluwer, Dordrecht.

Dan Klein and Christopher D. Manning. 2003. Accurate unlexicalized parsing. In *Proc. of the 41st Annual Meeting of the ACL*, pages 423–430.

W. McCune. 1998. Automatic proofs and counterexamples for some ortholattice identities. *Information Processing Letters*, 65:285–291.

Dan I. Moldovan, Christine Clark, Sanda M. Harabagiu, and Steven J. Maiorano. 2003. COGEX: A logic prover for question answering. In *Proc. of HLT/NAACL*, pages 87–93.

Marius Pasca and Sanda M. Harabagiu. 2001. High performance question/answering. In *Proc. of SIGIR*, pages 366–374.

Stanford NLP Group. 2004. Project website. http://nlp.stanford.edu/nlkr/.

David Warren and Fernando Pereira. 1982. An efficient easily adaptable system for interpreting natural language queries. *Computational Linguistics*, 8(3-4):110–122.

Karl Weber. 1999. *The Unofficial Guide to the GRE Test.* ARCO Publishing, 2000 edition.

W. A. Woods. 1973. Progress in natural language understanding: An application to lunar geology. In *AFIPS Conference Proceedings*, volume 42, pages 441–450.