Coping with Ambiguity in Knowledge-based Natural Language Analysis

Kathryn L. Baker, Alexander M. Franz, Pamela W. Jordan
Center for Machine Translation
and
Department of Philosophy
Carnegie Mellon University
klb@cs.cmu.edu

Abstract

This paper describes the strategies and techniques used by the English analysis component of the KANT Knowledge-based Machine Translation system to cope with ambiguity. The constraints for elimination of ambiguity are distributed across the various knowledge sources in the analyzer. As a result, efficiency in the analysis component is maintained, and output quality is improved.

1 INTRODUCTION

The KANT system [Nyberg and Mitamura, 1992] is a Knowledge-based Machine Translation (KBMT) system designed to translate English source documents into multiple target languages. The current application of the KANT system is the translation of service information publications for all major products of Caterpillar, Inc. (heavy equipment) into the major export languages.

One of the difficulties in implementing a KBMT system, as with many other natural language applications, is the ambiguity that arises during analysis. Previous knowledge-based natural language analysis systems have identified a number of different knowledge sources that are necessary to perform automatic disambiguation [Hirst, 1986, Goodman and Nirenburg, 1991]. In the KANT system, a number of knowledge sources have been integrated into a working system that operates over a circumscribed, but large domain.

The KANT system uses the "Universal Parser" [Tomita and Carbonell, 1987] with a grammar formalism based on Lexical-Functional Grammar [Bresnan, 1982]. The grammar consists of context-free rules that define the input's constituent structure (c-structure). The rules are annotated with constraint equations that define the input's functional structure (f-structure). Tomita's parser compiles the grammar into an LR-table and the constraint equations into Lisp code. Although this compilation results in fast parsing, the need to minimize ambiguity still exists. The generalized LR parser runs in polynomial time when the grammar has a minimal amount of ambiguity, but when the grammar is densely ambiguous it may take more than $O(n^3)$ [Tomita, 1986]. Fortunately, it is unlikely that natural language grammars will be so densely ambiguous as to take $O(n^3)$ to process. However, the less ambiguous the grammar, the faster the algorithm.

In addition to the parser, another integral part of the analysis component is a semantic domain model. In KANT, the relevant knowledge sources which contribute to the domain model are reorganized into data structures that are optimized for ambiguity resolution during parsing.

There are four main strategies for coping with ambiguity in KANT. The first strategy is to reduce ambiguity in the input text prior to language analysis. The second strategy is to incorporate preferences into the grammar when heuristics can be provided. The third strategy is to use semantic constraints provided by the domain model. The final strategy is to have the author resolve any remaining ambiguities. All of these strategies are important for improving the results of analysis. Although these strategies add some overhead, the main effect is to limit the number of parsing paths which improves the overall performance of analysis.

2 THE PROBLEM OF AMBIGUITY

Ambiguity arises in natural language analysis when more than one interpretation is possible for a given sentence. The ambiguity may be *lexical*, *structural*, or *semantic*. *Lexical ambiguity* occurs when a lexical entry allows a word more than one possible meaning. For example, in a general lexicon, the word *mat* might refer to *a flat article used for protection or support*. In the heavy equipment domain, however, the *mat* in the discourse might be instead *the layer or blanket of asphalt that is laid by a paving* *machine*. In the analysis of large corpora, the domain is key in determining the sense a word or phrase will take during analysis.

Syntactic ambiguity occurs when there are different possible syntactic parses for a grammatical sentence. An example of such ambiguity is the problem of *attachment* of modifiers to the proper constituents. Consider the sentence *Fasten the assembly with the lever.* This may be either an instruction to fasten the assembly using a lever, or an instruction to fasten the assembly, which has a lever attached to it. With the former interpretation, the prepositional phrase *with the lever* attaches to the verb, and with the latter, it attaches to the noun phrase object.

Semantic ambiguity refers to the broad category of ambiguity which arises when the meaning of the sentence must be determined with the help of greater knowledge sources. The problem of resolving simple pronominal reference is an example of semantic ambiguity. In the sentence *Start the engine and keep it running*, the fact that *it* refers to the engine is not inferrable from the single clause *keep it running*. Knowledge of the prior clause is necessary to resolve the pronoun.

3 CONSTRAINING AMBIGUITY IN THE SUBLANGUAGE

Given the size of the KANT domain, the inherent ambiguity in English, and the processing complexity of natural language analysis and translation, the strategy of constraining the input text via a sublanguage is an important way to reduce the overhead of the system. We also see benefits in a clear, consistent authoring style across authors and across document types. We constrain the input language by limiting the words and phrases in the lexicon to a single sense, and by restricting the syntactic constructions which are allowed in the controlled grammar.

3.1 Constraining the Lexicon

The KANT lexicon consists of single words and phrases. The problem of lexical ambiguity is addressed directly in the heavy equipment lexicon by limiting almost all general content words (e.g. *drain, right*) to one *sense* per part of speech. About 99% of the approximately 9,600 single-word lexical entries (general content words and single-word technical terms) are unambiguous. In the heavy equipment domain, *mat* has the definition *a layer or blanket of asphalt*. To guide the author in using a vocabulary item correctly, the user interface provides definitions and usage examples for each general content word, and offers synonymous alternatives for senses which are ruled out. For those words for which the sense remains ambiguous (approximately 100 so far in the heavy equipment domain), the author interacts with a *lexical disambiguator* during processing (see section 6).

The lexicon for the heavy equipment domain contains approximately 54,000 *nomenclature phrases* which, like the single word vocabulary, have been extracted from a large cross-section of the domain using *corpus analysis* [Mitamura et al., 1993]. These phrases each have a single sense in the lexicon. The author is encouraged to always select the most specific phrase available in the lexicon for the meaning that he or she wishes to convey. Thus, the author may choose e.g. either of the terms *floor mat* or *rubber mat* when the "mat" in the discourse is *a flat article used for protection or support*. Because of the rich phrasal vocabulary, the author is able to express himself adequately.

3.2 Constraining the Source Language Constructions

The syntax of the source text is constrained by a controlled grammar consisting of rules and recommendations. The rules and recommendations are in place to reduce both syntactic ambiguity and semantic ambiguity.

The rules of the controlled grammar list the syntactic structures which will be accepted by the parser. These are chosen to limit the amount of ambiguity resolution which will be necessary during processing. Simple declarative and imperative sentences are part of the controlled grammar, as is the conjunction of noun phrases, prepositional phrases, and sentences. The grammar also includes unambiguous relative clauses. Punctuation, e.g. the proper use of period, is strictly specified.

Other constructions are not part of the controlled grammar, because they lead to ambiguity. One example of a construction which is not allowed due to syntactic ambiguity is verb phrase conjunction. Consider the sentence *Stop and inspect the engine.* The verb *stop* may be intransitive, in which case *the reader himself stops*, or it may be transitive, in which case *the reader stops the engine*. Conjunction of the verbs *stop* and *inspect* is not part of the controlled grammar. Another constituent type which is not part of the controlled grammar is the pronoun. The sentence *Stop the engine and keep it running* is not part of the controlled grammar, because the referent of *it*, a pronoun, is not easily resolvable. This is an example of ruling out semantic ambiguity during authoring.

The recommendations in the controlled grammar include guidelines for how to rewrite a text from general English into the domain language. The recommendations are useful both for rewriting old text and for creating new text. An example of a rewriting guideline is the following:

Problematic Text: *Fasten the assembly with the lever.*

Suggested Rewrite: *Use the lever in order to fasten the assembly*.

4 PARSING PREFERENCES

To reduce ambiguity, the parsing grammar either needs to prevent a structure from being built for a given syntactic context, or prevent it from successfully participating in an analysis when a preferred structure is present. The first case is easily implemented with constraint functions but the second case is not easy to achieve. The difficulty is that the decision about whether to prefer one or another construction seldom occurs in one rule. An example of this is an input that is ambiguous between a passive and predicate adjective analysis: *A key is required in order to unlock the steering wheel*. In this example, *required* could be either a predicate adjective or the passive participle. So, for instance, when an adjective phrase is promoted to the status of a predicate adjective we must anticipate what information needs to be added to the f-structure. This information will be used to decide whether it should be a predicate adjective or whether it should be blocked in hopes that the passive analysis will survive.

The rule that makes the decision about the predicate adjective has no influence on the passive rule and so the passive rule must also separately decide to block or allow the passive reading. We chose to use heuristics to make these decisions instead of a multiple-pass parse, recording and checking the chunks that have been built so far, or probablistic weightings associated with entries in the LR table [Briscoe and Carroll, 1993] to order the analyses. In future work we hope to explore these other possibilities. The preference heuristics for the passive vs. predicate adjective analysis are:

```
In the Predicate adjective grammar rules:

If a PP attached to the adjective form can
attach to the verb form of the -ed word
  then block the predicate adjective
Elseif the verb form of the -ed word is an
action verb
  then block the predicate adjective

In the Passive grammar rules:

If the verb is a stative verb
  then block the passive
```

5 USING SEMANTICS

Certain types of ambiguity can only be resolved by using semantic information. Consider the following sentence: *Lift the engine with the beam.* In order to choose the correct attachment site for the Prepositional Phrase (PP) *with the beam*, the parser needs to consider the meaning of the verb *lift*, and the nouns *engine* and *beam*. This section describes a practical method for integrating semantic rules into the LR parser. The resulting system combines the merits of a semantic domain model with the generality and wide coverage of syntactic parsing, but is fast and efficient enough to remain practical.

5.1 The Domain Model

The semantic knowledge that is required to resolve ambiguities like PP-attachment is represented in the domain model. The domain model is implemented as an inheritance hierarchy. Possible attributes for concepts, along with semantic constraints on the fillers, are inherited through this hierarchy.

Concepts are represented as simple frames. A frame has a head, one or more ancestors in the hierarchy, and zero or more attributes that are restricted to certain fillers. For example, below is the frame for the concept corresponding to the verb *to lift*:

```
(*A-LIFT
  (instrument *O-BEAM))
```

5.2 Knowledge Reorganization

Disambiguating information from the domain model is applied at the earliest possible stage during parsing. Whenever the parser tries to perform an attachment, it calls a function that checks the domain model for semantic information that would license the attachment.

Mapping the f-structures to semantic concepts and searching the domain model at parse time would not lead to acceptable system performance. Instead, the *knowledge reorganizer* combines and transforms the semantic knowledge sources into a different data structure that is optimized for lookup during parsing. The resulting data structures are called the “semantic restrictors”. The knowledge reorganizer performs the following steps:

1. Read input files (lexical mapping rules, domain model frames, semantic interpretation rules).

2. Construct linked inheritance tree.

3. Perform inheritance of all semantic properties.

4. Build semantic restrictor for each noun and verb.

5. Introduce structure-sharing in semantic restrictors; Lisp-compile the structure-shared semantic restrictors.

6. Write is-a hierarchy in table format; Lisp-compile the “is-a” table.

5.3 Semantic Restrictors

The result of this process is a semantic restrictor for every noun and verb. The semantic restrictor describes all possible modifiers along with their semantic roles, and also includes patterns for modifiers that are not licensed for attachment. The structure of a semantic restrictor is as follows:

```
(<concept-head>
  (<syntactic-path>
    ({(<semantic-path>
       <semantic-filler>+) | FAIL}
     [syntactic-constraint>])+)+)
```

An example restrictor for the concept `*A-LIFT` is shown below.

```
(*A-LIFT
 ((PP OBJ)
  (FAIL
      ((PP ((ROOT (*OR* "in" "on" "at"))))))
  (FAIL
      ((PP ((ROOT (*OR* "to" "in"))))))
  ((INSTRUMENT *O-BEAM)
      ((PP ((ROOT (*OR* "with" "by"))))))
  (FAIL ((PP ((ROOT "than")))))
  (FAIL ((PP ((ROOT "near")))))
  (FAIL ((PP ((ROOT "about")))))
  (FAIL ((PP ((ROOT "from")))))
  (FAIL ((PP ((ROOT "for")))))
  ...
```

If we do not have any data for a new modifier, we can treat it as a special case. If we wish to increase coverage, we can allow such a modifier to attach as the filler of some default semantic role. On the other hand, we might only wish to allow attachment of modifiers that we know about – this might help to guarantee high accuracy during language analysis. In this case, we can deny attachment of all novel modifiers.

5.4 Parse-time Disambiguation

The reorganization of the information from the domain model is performed off-line. During parsing, only two tables are loaded: a table with the restrictor data, and a table that contains the transitive closure of the is-a relation. When an attachment grammar rule is fired by the parser, the following sequence of events occurs:

1. The attachment site and modifier are mapped from <*root,category*> to the domain model concept. The concept is stored in the f-structure for future reference.

2. The restrictor for the attachment site is looked up in the restrictor table.

3. The modifier f-structure is matched against the appropriate attachment slot in the restrictor.

4. If there is a match, the is-a table is consulted to check the semantic role-filler restrictions. If the restrictions are met, attachment is licensed. Otherwise, it is denied.

5. If there is no match, the modifier is not covered by the restrictor, and it is treated as a special case.

6 INTERACTIVE DISAMBIGUATION

When all other possibilities for automatically resolving an ambiguity are exhausted, the author is consulted. Our implementation of author disambiguation is similar in spirit to [Tomita, 1984]. The main difference is that we record the results of the disambiguation with SGML (*Standard Generalized Markup Language*) tags that are inserted into the original text. The interaction with the author is not at the level of guiding the parser during analysis as in [Briscoe and Carroll, 1993] but at the level of choosing the correct analysis once parsing is complete.

A majority of the unresolved structural ambiguities are PP-attachments. Some PP-attachment ambiguities still occur since the domain model may license it to attach to multiple sites in the sentence. Returning to an earlier example, if the author were to overlook the rewriting of *Fasten the assembly with the lever*, then *with the lever* is licensed to attach as an *instrument* to *fasten* and as a *has-as-part* to *assembly*.

Interactive ambiguity resolution is triggered when multiple f-structures are produced by the parser. The attachment sites for each word in the sentence are collected from the f-structure and when multiple sites are found, the ambiguity is presented to the author with information such as:

```
"What 'with' modifies is ambiguous."
AUTHOR CHOICE: "fasten with"
AUTHOR CHOICE: "assembly with"
```

The author is asked which of the sites the preposition, *with* for example, attaches to. The author's response is formatted into an SGML tag that gets inserted into the input sentence. The tag is necessary, since the text containing the ambiguous sentence may get edited at a later time or re-analyzed again when the source text is to be translated into a new target language. The sentence is then re-analyzed by the parser after the author's edits are complete. The SGML tag acts to constrain the attachment site to the one selected by the author. The tag inserted into the input sentence as a result of the author choosing *fasten with* appears as:

```
"fasten the assembly with<?CTE attach
head='fasten' modi='with'> the lever."
```

In the grammar, the tag first gets incorporated into the f-structure for the word that had multiple attachment sites and as a result indicates the word it is constrained to attach to. This grammar rule appears below:

```
(<term> <== (<term><disambig-tag>)
        ((x2 cat) =c attach)
        ((x2 modi) = (x1 root))
        (x0 = x1)
        ((x0 disambig) = (x2 head)))
```

The other grammar rules that involve the possible attachment sites each have constraint rules that check their modifiers for disambiguation constraints. When the constraints are present, the rule checks whether the attachment is allowed. An example of one such grammar rule is:

```
(<adj3> <== (<measurement-exp> <adj2>)
         ((*EOR*
           (((x1 disambig) = *defined*)
            ((x1 disambig) = (x2 root)))
           (((x1 disambig) = *undefined*)))
          (x0 = x2)
          ((x0 measure-adj-mod) = x1)))
```

Lexical ambiguity resolution requires just one grammar rule. With this type of ambiguity the author is asked to select the appropriate meaning from among the allowed domain senses. The tag indicating the selected word sense is allowed to combine only with the f-structure for the appropriate sense. Any f-structure with another sense will fail to combine with the tag and that parse will fail. The grammar rule for this is:

```
(<term> <== (<term><disambig-tag>)
        ((x2 cat) =c means)
        ((x1 sem) = *defined*)
        ((x2 sem) = (x1 sem))
        (x0 = x1)))
```

7 CONCLUSIONS

We have described a number of strategies for coping with ambiguity in the framework of knowledge-based natural language analysis. The strategies include ways of reducing ambiguity in the input text, and methods for resolving ambiguity during and after parsing. While we have not performed a formal analysis of the complexity of these strategies, our experience with the KANT system shows that the strategies are effective in reducing the number of parses without significantly degrading the performance of the analyzer. Compared with the gains in output quality, the performance penalty is minimal.

8 Acknowledgements

We would like to thank our colleagues on the KANT project, especially Jaime Carbonell (CMT director), Nicholas Brownlow, Sue Holm, Teruko Mitamura, and Eric Nyberg. We also thank Caterpillar, Inc., for whom the large heavy-equipment application of KANT has been constructed, and our colleagues at Carnegie Group Inc.

References

[Bresnan, 1982] Bresnan, J. W. (1982). *The Mental Representation of Grammatical Relations*. MIT Press, Cambridge, MA.

[Briscoe and Carroll, 1993] Briscoe, T. and Carroll, J. (1993). Generalized probabilistic LR parsing of natural language (corpora) with unification-based grammar. *Computational Linguistics*, 19(1):25–59.

[Goodman and Nirenburg, 1991] Goodman, K. and Nirenburg, S. (1991). *The KBMT Project: A Case Study in Knowledge-Based Machine Translation*. Morgan Kaufmann, San Mateo, CA.

[Hirst, 1986] Hirst, G. (1986). *Semantic Interpretation and the Resolution of Ambiguity*. Cambridge University Press, Cambridge.

[Mitamura et al., 1993] Mitamura, T., Nyberg, E., and Carbonell, J. (1993). Automated corpus analysis and the acquisition of large, multi-lingual knowledge bases for MT. In *5th International Conference on Theoretical and Methodological Issues in Machine Translation*, Kyoto, Japan.

[Nyberg and Mitamura, 1992] Nyberg, E. and Mitamura, T. (1992). The KANT system: Fast, accurate, high-quality translation in practical domains. In *Coling-92*.

[Tomita, 1984] Tomita, M. (1984). Disambiguating grammatically ambiguous sentences by asking. In *Coling-84*.

[Tomita, 1986] Tomita, M. (1986). *Efficient Parsing for Natural Language: A Fast Algorithm for Practical Systems*. Kluwer Academic Publishers, Boston, MA.

[Tomita and Carbonell, 1987] Tomita, M. and Carbonell, J. (1987). The universal parser architecture for knowledge-based machine translation. Technical Report CMU-CMT-87-101, Center for Machine Translation, Carnegie Mellon University.