SYNTACTIC/SEMANTIC COUPLING IN THE BBN DELPHI SYSTEM

Robert Bobrow, Robert Ingria, David Stallard

BBN Systems and Technologies
10 Moulton Street
Cambridge, MA 02138

ABSTRACT

We have recently made significant changes to the BBN DELPHI syntactic and semantic analysis component. These goal of these changes was to maintain the tight coupling between syntax and semantics characteristic of earlier versions of DELPHI, while making it possible for the system to provide useful semantic interpretations of input for which complete syntactic analysis is impossible. Semantic interpretation is viewed as a process operating on a sequence of messages characterizing local grammatical relations among phrases, rather than as a recursive tree walk over a globally complete and coherent parse tree. The combination of incremental semantic interpretation and statistical control of the parsing process makes it feasible to reconstruct local grammatical relations with substantial accuracy, even when a global parse cannot be obtained. Grammatical relations provide the interface between syntactic processing and semantic interpretation, and standard global parsing is viewed as merely one way to obtain evidence for the existence of such grammatical relations in an input string. This focus on grammatical relations leads to substantial simplification of both grammar and semantic rules, and will facilitate our ultimate aim of acquiring syntactic, semantic and lexical knowledge by largely automatic means.

1. THE PROBLEM

There are two long standing problems of computational linguistics for systems which do syntactic processing before semantic processing. First, they are limited by the coverage of their grammar. Second, "syntactically ill-formed" word sequences, which represent a noticeable fraction of the utterances in any natural setting (e.g. spontaneously spoken input or unedited text) cause many failures. Other architectures have their own problems. Systems that depend primarily on semantic processing tend to be uncomfortably domain-dependent, working best in highly constrained problem domains. "Semantic grammar" systems often do not capture a wide range of syntactic variations with the same meaning, while "frame based" systems typically allow for ill-formedness and syntactic variation by forcing all input into the procrustean bed of a highly limited task model.

2. DELPHI'S APROACH

Our goal in the DELPHI system has to develop techniques that allow general, task independent, syntactic knowledge to be used to the fullest extent possible, making it feasible to encode semantic knowledge in the simplest (and thus most learnable) form, without sacrificing generality. Classical approaches to this ideal fail completely when presented with "syntactically ill-formed" input, whether that ill-formedness is due to the system's incomplete representation of syntactic regularities, or to genuine disfluencies on the part of the speaker/writer. We have been continually making progress toward a balanced approach that allows us to take advantage of syntactic constraints wherever possible, while allowing the system to interpret inputs for which no grammatical parse can be found.

The differences between our approach and more standard syntactically oriented approaches are subtle. At first glance, our grammar and our parser do not look radically different than thosed used in other syntactic analysis approaches (with the exception of the scheduling algorithm mentioned below).

We started with a relatively standard context-free parsing algorithm, applied to what is for the most part a straightforward unification-based grammar. The largest modification to the parser was its conversion to an agenda-based chart-parser, with scheduling depending on measured statistical likelihood of grammatical rules [1]. This enhanced efficiency significantly, by allowing us to generate parses in a "best first" order, but did not change the syntactic coverage of our system.

All versions of DELPHI for the last several years have integrated semantic processing with parsing. This ensures that all syntactic structures placed in the chart are semantically coherent, further reducing the search space for the best parse. In the early versions of DELPHI, each syntactic rule had an associated semantic rule which had to be sucessfully applied before the syntactically hypothesized constituent would be accepted in the chart. Because of the large number of syntactic rules needed to have a broad coverage grammar, the number of semantic rules was quite large, and the representation for lexical semantics was quite complex.

2.1 Parsing as Transduction

The biggest change in DELPHI came as we started to look at the parser, not as a device for constructing syntactic trees, but as an information transducer that makes it possible to simplify and generalize the rules for semantic interpretation. The purpose of syntactic analysis in this view is to make information encoded in ordering and constituency as readily available as the information encoded in the lexical items, and to map syntactic paraphrases to informationally equivalent structures. The actual interface between parsing and semantics is a dynamic process structured as a cascade (as in Woods notion of cascaded ATNs [4]), with parsing and semantic interpretation acting as coroutines. The input to the semantic interpreter is a sequence of messages, each requesting the "binding" of some constituent to a head. The semantic interpreter does not perform any sort of recursive tree-walk over the syntactic structure produced by the parser, and is in fact immune to many details of the tree structure.

This view of a grammar as a transducer between input strings and semantic representation made it possible for us to substantially restructure the grammar in such a way as to both decrease the number of rules and increase its coverage. The original DELPHI grammar contained 1143 rules. The restructured grammar has only 453 rules. This overall number perhaps underestimates the impact of the change in point of view, because it includes rules for various specialized subgrammars such as numbers, latitudes and longitudes and clock times, which were not revised. The number of VP rules (excluding conjunction and modal rules) dropped from 83 to 15, while the coverage of VP phenomena increased.

2.2 The "Piece Parts" Metaphor

In general, while certain orderings of modifiers seem strongly constrained by grammar (determiner and relative clause for NPs, subject and object for clauses, indirect object and object for VPs), other orderings seem to be more weakly determined (the "arguments" of a verb, such as the origin and destination phrases of a verb of motion, usually occur before more general verbal adjuncts like time-modifiers), and can be over-ridden by factors such as such as "heaviness". Thus, most attachments can be modelled by simple binary adjunction. Since the exact topology of the parse tree could be modified without materially affecting the transduction operation, we opted to generate complex recursive structures primarily by left and right adjunction, using rules of the general form

```
(X ...)  => (X-LEFT-MOD ...)  (X ...)
```

and

```
(X ...)  => (X ...)  (X-RIGHT-MOD ...)
```

When the structures produced by such rules are written in a bracketed notation the resulting items look like nothing so much as onions!

```
((((X...)  (X-RIGHT-MOD ...))  ...)...)
...)
```

The critical issue for a transducer, however, is the information flow from syntax to semantics, and at this level the layers of the onion disappear, with each adjunct being "logically attached" to the "head" of the constituent.

In effect, we have factored a number of constructions that were previously treated as units into "piece parts" that can be combined together in various ways, subject to semantic well-formedness. Verb subcategorization is a prime example of one such area [2]. Rather than using subcategorization features to name sets of categories that appear together as complements, we have defined approximately 15 verb phrase rules that list the possible constituents that may appear as complements to a verb. These may embed within each other freely, so long as the results are semantically interpretable by the head.

We have adopted this approach throughout the grammar. For example, the complements and adjuncts that may appear within a noun phrase are introduced by recursive NP rules, similar to the VP rules we have discussed here. This recursive scheme allows the piece part rules of the grammar to be combined together in novel ways, governed by the lexical semantics of individual words. The grammar writer does not need to foresee all possible combinations, as before.

2.3 Examples

For example, in an earlier version of the grammar, in order for a verb, like "fly", to take two prepositional phrase complements, the following rule was required (for the purposes of exposition we suppress complex unification structures) :

```
(VP ...  :SUBJ :WFF) =>
(V :WORD ...
(DITRANSPREP :PREP :PREP1 ...  :PP1
:PP2) ...)
(PP :PREP ...  :PP1)
(PP :PREP1 ...  :PP2)
```

The word "fly" contained the feature (DITRANSPREP (FROMPREP) (TOPREP) ...) in its lexical entry to constrain the prepositions to be "from" and "to". In order for "fly" to take the prepositions in the opposite order, as well, either the lexical entry for "fly" would have contained the additional subcategorization entry (DITRANSPREP (TOPREP)

(FROMPREP) ...) with the values of the two preposition arguments reversed, or we would have needed to add the following "lexical redundancy rule" to the grammar:

```
(VP ...  :SUBJ :WFF) =>
(V :WORD ...
(DITRANSPREP :PREP :PREP1 ...  :PP1
:PP2) ...)
(PP :PREP1 ...  :PP2)
(PP :PREP ...  :PP1)
```

which automatically inverts the order of :PREP and :PREP1 for any verb taking two prepositional phrases. To allow "fly" to take a "to" phrase without a "from" phrase, as often occurs, we would need both another subcategorization specification in the lexical entry for "fly", and another rule in the grammar, allowing a verb to take a single PP complement.

In the current grammar, all PP complements are handled by one single rule:

```
(VP :AGR ...)  =>
:HEAD (VP :AGR ...)
:PP-COMP (PP :PREP ...)
```

This rule allows a verb to take a single prepositional phrase complement, or, by recursion, an indefinite number of others, consistent with its semantics. The lexical information particular to individual verbs governs the number of prepositions that the verb takes, their optionality or obligatoriness, and their semantic interpretation. Special purpose rules for different numbers and orders of PPs are not required.

3. GRAMMATICAL RELATIONS

The chief effort over the last year has been to codify this notion of logical attachment, simplifying the set of such attachments to highlight the common underlying substructure of grammatical paraphrases. To this end we re-oriented our grammar around the notion of "grammatical relation". Grammatical relations include the familar ones of deep-structure subject and object, as well as other relations. These relations may be seen as the end result of making the information encoded in ordering and constituency explicitly available. (In languages with freer word order this information is often encoded in morphological affixes or pre and post positions.) From the point of view of a syntactic-semantic transducer, the key point of any grammatical relation is that it licenses (one of) a small number of semantic relations between the ("meanings" of) the related constituents. Sometimes the grammatical relation constrains the semantic relation in ways that cannot be predicted from the semantics of the constituents alone (given "John", "Mary" and "kissed", only the grammatical relations or prior world knowledge determine who gave and who received). Other times the grammatical relation simply licenses the one plausible semantic relation (given "John", "ate" and "hamburger", if there is a relation, it is the hamburger that is most likely to have been consumed—but in the sentence "John ate the fries but rejected the hamburger" our knowledge of the destiny of the hamburger is mediated by its lack of a grammatical relation to "ate").

Grammatical relations are incorporated into the grammar by giving each element of the right hand side of a grammar rule a grammatical relation as a label. Some typical rules are, in schematic form:

```
(NP ...)  =>
:HEAD (NP ...)
:PP-COMP (PP :PREP ...)
```

```
(N-BAR ...)  =>
:PRE-NOM (N ...)
:HEAD (N-BAR ...)
```

The first indicates that an NP may have an NP as a head and a PP as an adjunct, with the grammatical relation :pp-comp holding between them (the actual operation of binding a :pp-comp splits it into a number of sub-relations based on the preposition, but that can be safely ignored here). The second indicates that the head need not occur as the first constituent on the right side. All that is required is that one of the right-hand elements is labeled as the "head" of the rule, and it is the source of information about the initial semantic and syntactic "binding state". This binding state controls whether or not the other elements of the right-hand side can "bind" to the head via the relation that labels them. Semantics is associated with grammatical relations, not with particular grammar rules (as are Montague-grammar and most unification-based semantics systems).

3.1"Binding rules" – the Semantics of Grammatical Relations

The implementation of the use of such binding states in the transduction of grammatical relations to semantic structure is facilitated by the procedural elements we have introduced into our unification grammar formalism. In early versions of DELPHI, grammar rules contained only unification expressions for grammatical constituents. Later versions added "logical nodes"—expressions which looked like constituents, but which were satisfied by deductions in a unification-based axiom set. These logical nodes were used to add various constraints to the grammar, much as in a definite clause grammar. An analysis of the time spent in

parsing showed that substantial time was spent in such logical computations, and it became clear that more efficient data structures and procedural techniques could be used to implement many such computations [3]. The current version of the system uses these embedded procedural mechanisms to manipulate specialized data structures that efficently represent the binding state of a constituent, to determine if a proposed grammatical relation leads to a consistent binding state, and if so what the semantic implications of that binding are.

A separate system of "binding rules" for each grammatical relation licenses the binding of a constituent to a head via that relation by specifying the semantic implications of binding. These rules generally specify aspects that must be true of the semantic structure of the head and bound constituent in order for the binding to take place, and may also specify certain syntactic requirements. They may take into account the existence of previous bindings to the head, allowing certain semantic roles (such as time specification) to be filled multiply, while other semantic roles may be restricted to having just one filler.

As adjuncts are added to a structure the binding list is extended. As layers are added to the onion, a simple linear list of bindings is maintained representing the head and its grammatical relation to each of the constituents added with each layer of the onion. Semantic binding rules are used to verify the local semantic plausibility of a structure, i.e. the semantic plausibility of each proposed grammatical relation. The next phase of semantic interpretation takes place when the onion is complete, i.e. when a constituent X is inserted as other than the head of a larger constituent. This situation provides evidence that the outermost layer of the onion has been reached, and that no more adjuncts are to be added. At this time it is possible to evaluate semantic rules that check for completeness and produce an "interpretation" of the constituent. These completion rules operate directly on the binding list, not on the recursive left or right branching tree structure produced by direct application of the grammar. The actual tree structure is at this level immaterial, having been replaced by the flattened binding list representation of relational structure.

4. ROBUSTNESS BASED ON STATISTICS AND SEMANTICS

Simply having a transduction system with semantics based on grammatical relations does not deal with the issue of robustness - the ability to make sense of an input even if it cannot be assigned a well-formed syntactic tree. The difficulty with standard syntactic techniques is that local syntactic evidence is not enough to accurately determine grammatical relations. A NP (e.g. "John") followed by a verb (e.g. "flew") may be the subject of that verb (e.g. "John flew to Boston") or may be unrelated (e.g. "The man I introduced to John flew to Boston"). The standard way of getting around this is to attempt to find a globally consistent set of grammatical relation labels (i.e. a global parse!) and make use of the fact that the existence of a global parse containing a given relation is stronger evidence for that relation than local structure (although syntactic ambiguity makes even such global structures suspect). This is indeed the best approach if all you have available is a syntactic grammar.

The strategy we use in DELPHI is based on the existence of two other sources of information. In the first place we have semantic constraints that can be applied incrementally, so that we can check each proposed grammatical relation for semantic coherence in the context of other assumed grammatical structures. Additionally, we have statistical information on the likelihood of various word senses, grammatical rules, and grammatical-semantic transductions. Thus we can not only rule out many locally possible grammatical relations on the basis of semantic incoherence, we can rank alternative local structures on the basis of empirically measured statistics. The net result is that even in the absence of a single global parse, we can be reasonably sure of the local grammatical relations and semantic content of various fragments (we can even give numerical estimates of the likelihood of each such structure).

4.1 Control structure

The DELPHI system attempts to obtain a complete parse of its input, using its agenda-based best-first parsing algorithm. If it is unable to do this it uses the parser in a fragment-production mode, producing the most probable structure for an initial segment of the input, then restarting the parser in a top down mode on the first element of the unparsed string whose lexical category provides a reasonable anchor for top-down prediction. This process is repeated until the entire input is spanned with fragments. Experiments have shown that the combination of statistical evaluation and semantic constraints lets this procedure produce a highly useful chunking of the input for interpretation by other non-syntactically driven strategies. Further details are given in the accompanying paper on the DELPHI fall-back processing strategies.

5. ADVANTAGES OF THIS APPROACH

The separation of syntactic grammar rules from semantic binding and completion rules has important consequences for processing. First, it enables the notion of grammatical relation to be separated from the notion of tree structure, and thus greatly facilitates fragment parsing. Second, while it allows syntax and semantics to be strongly coupled in terms of processing (parsing and semantic interpretation) it allows them to be essentially decoupled in terms of notation. This makes the grammar and the semantics considerably easier

to modify and maintain.

We believe, however, that in the long term the most important advantage is that this view leads us to a new kind of language model, in which knowledge can be much more easily extracted through automatic training. We view the role of the grammar as codifying the way that tree structure provides evidence for grammatical relations. Thus the rule

```
(NP ...)  =>
:HEAD (NP ...)
:PP-COMP (PP :PREP ...)
```

says that a noun phrase followed by a prepositional phrase provides evidence for the relation PP-COMP between the PP and NP head.

The separation between rules types will allow us for the first time to consider the effect of grammatical relations on meaning, independently of the way that evidence for these relations is produced by the parser. One effect of this is to make it possible to use a hypothesized semantic interpretation of a set of tree fragments to generate a new syntactic rule.

Thus, in normal operation, the primary evidence for a grammatical relation is the result of actually parsing part of an input. However, since grammatical relations between constituents entail semantic relations, if we can make an estimate of the likelihood of certain semantic relations based on domain knowledge, pragmatics, and task models, etc., it is in principle possible to use abductive reasoning to suggest likely grammatical relations, and thereby propose new grammar rules. In effect, grammatical relations form an abstract level of representation that greatly simplifies the interaction of syntactic and semantic processing.

ACKNOWLEDGEMENTS

The work reported here was supported by the Advanced Research Projects Agency and was monitored by the Office of Naval Research under Contract No. N00014-89-C-0008. The views and conclusions contained in this document are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of the Defense Advanced Research Projects Agency or the United States Government.

REFERENCES

1. Bobrow, R. "Statistical Agenda Parsing", in *Speech and Natural Language: Proceedings of a Workshop Held at Pacific Grove, California, February 19–22, 1991*, Morgan Kaufmann Publishers, Inc., San Mateo, California, pp. 222–224.
2. Bobrow, R., R. Ingria, and D. Stallard) (1991) "The Mapping Unit Approach to Subcategorization", in *Speech and Natural Language: Proceedings of a Workshop Held at Pacific Grove, California, February 19–22, 1991*, Morgan Kaufmann Publishers, Inc., San Mateo, California, pp. 185–189.
3. Bobrow, R. and L. Ramshaw (1990) "On Deftly Introducing Procedural Elements into Unification Parsing", in *Speech and Natural Language: Proceedings of a Workshop Held at Hidden Valley, Pennsylvania, June 24–27, 1990*, Morgan Kaufmann Publishers, Inc., San Mateo, California, pp. 237–240.
4. Woods, W. (1980) "Cascaded ATN Grammars", in *American Journal of Computational Linguistics, January-March 1980, vol 6, no. 1*, Association for Computational Linguistics, p1-12.