THE SEMANTIC LINKER - A NEW FRAGMENT COMBINING METHOD

David Stallard and Robert Bobrow

BBN Systems and Technologies, Inc.
70 Fawcett St.
Cambridge, MA 02138

ABSTRACT

This paper presents the Semantic Linker, the fallback component used by the the DELPHI natural language component of the BBN spoken language system HARC. The Semantic Linker is invoked when DELPHI's regular chart-based unification grammar parser is unable to parse an input; it attempts to come up with a semantic interpretation by combining the fragmentary sub-parses left over in the chart using a domain-independent method incorporating general search algorithm driven by empirically determined probabilities and parameter weights. It was used in the DARPA November 92 ATIS evaluation, where it reduced DELPHI's Weighted Error on the NL test by 30% (from 32% to 22%).

1. INTRODUCTION

An important problem for natural language interfaces, as well as for other NL applications such as message processing systems, is coping with input which cannot be handled by the system's grammar. A system which depends on its input being grammatical (or on lying within the coverage of its grammar) simply will not be robust and useful. Some sort of "fallback" component is therefore necessary as a complement to regular parsing.

This paper presents the Semantic Linker, the fallback component used by the the DELPHI natural language component of the BBN spoken language system HARC. The Semantic Linker is invoked when DELPHI's regular chart-based unification grammar parser is unable to parse an input; it attempts to come up with a semantic interpretation by combining the fragmentary sub-parses left over in the chart. It was used in the DARPA November 92 ATIS evaluation, where it reduced DELPHI's Weighted Error on the NL test by 30% (from 32% to 22%).

The Semantic Linker represents an important departure from previous proposals, both our own [1] and others [2], in that it casts fragment combination as a general search problem, rather than as a problem of task model template matching (as in [4]) or as an extension to the existing parsing algorithm (as in [3]). Rather than reconstruct a parse tree, the goal of the search is to combine all the fragments into the most minimal and plausible connected graph, in which the links are not syntactic descendancy, but logical binary relations from the domain, such as "AIRLINE-OF", "ORIG-OF" etc.

States in the search space are partial connections of the fragments: in other words, a set of links. There a two types of "move" to reach a new state from an existing one. One adds a new link between fragments, and the other "hallucinates" an object to bridge two fragments that could not otherwise be linked (corresponding roughly to a notion of ellipsis). A success terminal state is one in which all the fragments have been linked. States have features associated with their constituent links and a system of weights on the features determines a score that is used to guide the search.

The advantages of this formulation are its domain-independence, flexiblity, extensibility, and ability to make use of statistical data. In particular:

- No assumption need be made about constraining task models
- The state space can be searched in any order
- New features are straightforward to add
- Probabilities of relations determined from (parseable) corpora can be used
- Weights on features are potentially derivable by automatic training

In the next sections we turn to a more detailed description of data structures and algorithms. We first give some necesary background on semantic interpretation in the DELPHI system, and on the generation and interpretation of fragmentary sub-parses in it. Next, we show how this framework is used to generate all possible connections between pairs of different fragment objects, and how probabilities and other features are assigned to these connections. We then show how we efficiently search the space of combinations of such links in order to find the minimal and plausible set of connections, and how such link combinations are turned into final interpretations. Finally, we give quantitative results, and discuss our future plans.

2. SEMANTIC INTERPRETATION OF FRAGMENTS

The central notion in DELPHI's syntactic-semantic interface is the "grammatical relation". Grammatical relations include the familar deep-structure complement relations of subject, direct-object etc., as well as other various adjunct relations, such as PP-COMP in the rule below:

```
(NP etc.)
->
HEAD (NP etc.)
PP-COMP (PP etc.)
```

The special grammatical relation "HEAD" denotes the head of the phrase. All other grammatical relations are said to "bind" a constituent they label – their "argument" – to this head to make a new object of the same category as the head. Here, a PP argument is bound to an NP head to make a new NP.

Binding operates on the semantic interpretation and subcategorization information of the head and on the semantic interpretation of the argument to produce the semantic interpretation of the new phrase. In principle, the relationship between inputs and output is completely arbitrary. In practice, however, it most often consists of an addition of a pair (RELATION, ARG-INTERP) to what are termed the "bindings" of the head input. For example, in the case of "flight on Delta" the pair added would be

```
(FLIGHT-AIRLINE-OF, DELTA)
```

In everything that follows, we will make this simplifying assumption.

We can then speak of a translation R → r from a grammatical relation to a semantic relation. For the present example, this translation would be:

```
PP-COMP(ON) -> FLIGHT-AIRLINE-OF
```

where the grammatical relation PP-COMP is further subdivided by the preposition "ON" (and the requirements on semantic type are implicit from the relation FLIGHT-AIRLINE-OF). We will term such a translation a "realization rule" because it shows how the semantic relation FLIGHT-AIRLINE-OF can be syntactically realized in terms of an on-PP. The set of all such realization rules (large in number for a non-trivial domain) is stored in a knowledge base separate from the parser and interpreter code.

The interpretation of any parse tree can now be represented as an isomorphic semantic tree, in which the nodes are the semantic interpretation objects of open-class lexical items and the links are the semantic relations between them. Such a structure can obviously also be represented as a set of n semantic objects and n-1 triples consisting of a semantic relation and head and argument semantic objects. For example, "Delta flies a 747 to Denver" would be represented in graph form as:

```
  /-------AIRCRAFT-OF -> 747
FLY---- AIRLINE-OF -> DELTA
  \-----DEST-OF -> DENVER:TO
```

where a PP such "to Denver" is represented as its NP object tagged by the preposition.

When a complete parse of an utterance cannot be performed, we are left with a set of fragmentary analyses in the chart which correspond to constituent analyses of portions of the input string. The Fragment Generator (essentially the same as was reported on in [1]) extracts the most probable fragment sub-parses associated with the longest sub-strings of the input, using probabilities associated with the producing grammar rules (as in [5].

The semantic interpretations of the parse-fragments are treated in the same way as those of a complete parse: as a set of objects and triples. As a simple example, suppose we have the three fragments "to Boston", "Denver" and "Delta flights on Monday". Then the three corresponding sub-graphs are:

```
BOSTON:TO

DENVER

FLIGHTS1------AIRLINE-OF ->  DELTA
    \--------- DAY-OF-WK -> MONDAY:ON
```

The problem of connecting the N fragments is then reduced to finding a set of relation-links which will connect a pair of objects in N-1 different fragments.

3. COMPUTING THE LINKS AND THEIR PROBABILITIES

The Semantic Linker first computes the link database, which is the set of all possible links between all pairs of objects in all pairs of different fragments. These links are computed using the same set of realization rules that drive the parser and semantic interpreter, and depend on the semantic types of the two objects and on the preposition tag (if any) of the second object. For the set of fragments in our example the link database is:

```
1a.   FLIGHTS1--- DEST-OF -> BOSTON:TO
1b.   FLIGHTS1--- ORIG-OF -> BOSTON:TO

2a.   FLIGHTS1--- DEST-OF -> DENVER
2b.   FLIGHTS1--- ORIG-OF -> DENVER

3a.   DENVER--- NEARBY-TO -> BOSTON:TO
```

where the links are grouped together in a ordered list according to the fragment-pairs they connect. Since there are three fragments there are three pairs.

Links have a set of features which are established when they are computed. The most important is the relational probability of the link, or:

```
P(r,C1,C2)
```

where r is the semantic relation of the link and C1 and C2 are semantic classes of the two argument positions, where C2 may be tagged by a preposition. This is the probability that a pair of objects of type C1 and C2 are linked by by the relation r in an interpretation (as opposed to by some different relation or by no relation at all).

A corpus of interpretations generated by hand could be used to determine these probabilities, but in our work we have chosen to work with a set of sentences that can be correctly parsed by the regular DELPHI parser. Since the semantic interpretations of these parses are just sets of triples the probabilities can be determined by counting. Approximately 3000 interpretations are currently used for our work in ATIS.

From this corpus, we can determine that the link 1a has a high (.89) probability of connecting a FLIGHT and CITY:TO object when these are present, whereas the link 3a has a near zero probability, since the relation NEARBY-CITY-OF occurs very infrequently between two cities.

We have found it convenient to use the log of these probabilities, scaled up and rounded to the lowest negative integer, as the actual value of the link probability feature. Additionally, maximum and minimum values of this number are imposed, so that even a highly likely link has a small negative score (-1), and a highly unlikely link has a finitely negative one (-70).

Links can have other features depending on assumptions made in computing them. For example, a link can be computed by ignoring the prepositional tag of the second object, in which case the link is given the feature "IGNORES-PREP". An example would be 1b above, which ignores the preposition "to". A link can also be computed by assuming a prepositional tag that is not present, giving the link the feature "ASSUMES-PREP", as in 3a, where the preposition "near" is assumed. As we shall see in the next section, these features are also assigned negative integers as penalties, balancing out any higher relational probability the link may have gained from the assumptions made by it.

4. SEARCHING THE SPACE OF COMBINATIONS

The problem of finding a connection between the N fragments is simply the problem of picking at most one link from each of the link-groups in the link database, subject to the constraints that all N fragments must be linked and that no links can be redundant.

We can formalize these constraints as follows. Let LINKED be defined as holding between two fragments if there is a link between them (in either direction), and let TC(LINKED) be the transitive closure of this relation. Then the first constraint is equivalent to the requirement that TC(LINKED) hold between all different fragments F1 and F2.

To formalize the non-redundancy constraint, let LINKED-L mean "linked except by link L". Then the non-redundancy constraint holds if there is no link L such that TC(LINKED) is the same as TC(LINKED-L).

The problem as cast implies a search space in which each state is simply the set of links chosen so far, and a transition between states is the addition of a new link. We will find it convenient, however, to include all of the following components in a state:

1. suffix of the link-database list
2. chosen-links
3. combinational features
4. state score
5. fragments-linked

The suffix of the link-database list consists of just the link-groups still available to be chosen. The combinational features are those arising from the combination of particular links, rather than from individual links themselves. The state score is the judgement of how plausible the state is, based on its features and those of its links. We want to find the most plausible success state, where a success state is one which satisfies the constraints above, as recorded on the fragments-linked slot.

Pre-success states reside on the state queue. The state queue initially consists of just the single state START. START has a pointer to the complete link-group list, an empty set of combinational features and links chosen and a score of zero. Search proceeds by selecting a state from the queue, and calling the function EXPAND-STATE on it to produce zero or more new states, adding these to the state queue and repeating until suitable success states are found or the queue becomes empty. Although this formulation allows the state space to be searched in any order, our implementation normally uses a best-first order choice. This simply means that at selection cycle, the best pre-success states are chosen for expansion.

The function EXPAND-STATE works by taking the first link-group from the link-group list suffix whose fragments are not already indirectly connected by the state and generating a new state a new state for every link L in the link-group. The links-chosen of these new states are the links-chosen of the parent state plus L, and the link-group suffix is the remainder of the parent's link-group suffix. EXPAND-STATE also generates a single new state whose link-group list suffix is the remainder but whose links-chosen are just those of the parent. This state represents the choice not to directly connect the two fragments of the link-group, and is given the feature "SKIP".

In our example, the first call to EXPAND-STATE would generate three new states from START: state S1 having the set {1a} as chosen-links, a state S2 having the set {1b} as its chosen-links and a state S3 having the empty set {} as its chosen-links, and the feature-list {SKIP}.

The score of a state is determined by summing the weighted values of its features and the features, including the log-probabilities, of its chosen links. Since the weights and log-probabilities are always negative numbers, the score of a state always decreases monotonically from the score of its parent, even in the case of a SKIP state.

At this point in our example, the state S1 has the best score, since its probability score is good (-2) and it has no "blemish" features, unlike the state S2, whose link 1b has the IGNORES-PREP feature. The SKIP state S3 is also not as good as S1, because the weight assigned to SKIP (-7) is selected so as to only be better than a link whose probability is lower than .50.

Thus, the state S1 is selected for expansion, resulting in the states S1-1, S1-2 and S1-3. The feature "CLASH", which results when a link with single-valued R (R a b) is combined with a link (R a b'), is assigned to S1-1, because it assigns the link 2a on top of 1a. The state S1-2 assigns the link 2b, which does not involve a clash. Both S1-1 and S1-2 are sucess states, and are therefore not expanded further.

Search then returns to the SKIP state S3. Its children all have lower scores than the success state S1-2, however, and given the guarantee that score decreases monotonically, any eventual success states resulting from them can never be as good as S1-2. They are therefore pruned from the search. The same happens with the descendants of other expansion candidates. The queue then becomes empty, and the best success state S1-2 is chosen as the result of fragment combination.

4.1. Hallucination

Suppose that instead of the example we have an utterance that does not include the word "flights":

Boston to Denver on Monday Delta

This utterance generates the fragments "Boston", "to Denver", "on Monday" and "Delta". Clearly, no complete set of links can be generated which would fully connect this set, without an object of semantic class FLIGHT or FARE to act as a "hub" between them.

To handle these situations, the Semantic Linker has a second type of state transition in which it is able to "hallucinate" an object of one of a pre-determined set of clases, and add link-groups between that hallucinated object and the fragment structures already present. In the ATIS domain, only objects of the classes FLIGHT, FARE, and GROUND-TRANSPORTATION may be hallucinated.

The hallucination operation is implemented by the function EXTEND-STATE. It is invoked when the function EXPAND-STATE returns the empty set (as will happen when input state's link-group list is empty) and returns states with the new link-groups added on, one for each of the allowed hallucination classes. These states are assigned a feature noting the hallucination, sub-categorized by the semantic class of the hallucinated object. Different penalty weights are associated with each such sub-categorized feature, based on the differences between probability of occurence of the classes in corpora. In ATIS, FLIGHT hallucinations are penalized least of all, FARE hallucinations more, and GROUND-TRANSPORTATION hallucinations most of all.

A state descended from one extended by hallucination cannot be extended again, and if it runs out of link-groups before connecting all fragments it is declared "dead" and removed from the queue.

4.2. Handling Corrections and Other Features

Several other combinational features influence the actions of the Semantic Linker with respect to such matters as handling speaker corrections and judging appropriate topology for the graph being built.

Speaker corrections are an important type of disfluency:

Tell me the flights to Denver uhh to Boston

This will produce the fragments "Tell me the flights to Denver" and "to Boston". Since a flight can have only one DEST-OF the fragment "to Boston" can not be connected as is. One strategy might be to ignore the "to" preposition and attempt to link "Boston" as an ORIG-OF with the IGNORE-PREP feature.

This clearly would not produce the correct interpretation, however. The Linker provides an alternative when the clashing value is to the right of the existing value in the string. In this case, the link receives the combinational feature REPLACEMENT, which is not penalized strongly. If the relational probability of the DEST-OF link is good, it will defeat its IGNORE-PREP rival, as it should.

Related to correction is the operation of *merging*, in which two nodes of a common semantic type are merged into one, and the appropriate adjustments made in the link-database and links-chosen for the state. This is appropriate for certain semantic classes where it is unlikely that separate descriptions (unless they are combined in a conjunction) will appear in an interpretation for the utterance:

Show me flights to Boston flights to Boston at 3 pm

Another feature influences the topology of the graph the Linker constructs. Nothing in the algorithm so far requires that graph structure of connections ultimately produced remain a tree, even though the input fragment interpretations themselves are trees. It is perfectly possible, in other words, for there to be two links (R a b) and (R' a' b) in which the same node is shared by two different parents.

Since we are not trying to produce a syntactic structure, but a semantic one in which the direction of relations is often irrelevant, we do not forbid this. It is discouraged, however, since it sometimes indicates an inapproriate interpretation. The combinational feature MULTI-ROLE is assigned to a state with such a combination of links, and is penalized.

Finally, we point out that the log-probability perspective is useful for assigning penalties to features. If one has a link L1 that has a high relational probability but also has a penalty feature, and another link L2 with a lower relational probability but which does not have the penalty, one can decide how far apart in probability they would have to be for the two alternatives to balance – that is, to be equally plausible. The difference in log-probabilities is the appropriate value of the penalty feature.

5. AFTER COMBINATION

After the combination phase is complete, we have zero or more success states from which to generate the utterance interpretation. If there are zero success states, an interpretation may still be generated through the mechanisms of "scavenging" and "back-off".

The Linker will find no success states either because it has searched the state-space exhaustively and not found one, or because pre-set bounds on the size of the space have been exceeded, or because the scores of all extensible frontier states have fallen below a pre-established pruning score for plausibility. In this case, the state-space which has been built up by the previous search is treated as an ordinary tree which the Linker scans recursively to find the optimum partial connection set, both in terms of fragment-percentage covered and in state score. This technique is termed "scavenging".

In some instances there may not even be partial connection states in the space. In this case, the system looks for the longest fragment to "back off" to as the interpretation.

In formal evaluation of the DELPHI system conducted under DARPA auspices[6], both scavenging and back-off were aborted in cases where there were obviously important fragments that could not be included in interpretation. This was done because of the signifigant penalty attached to a wrong answer in this evaluation.

If there is more than one success state, the Linker picks the the subset of them with the highest score. If there are more than a certain pre-set number of these (currently 2), the Linker concludes that it none of them are likely to be valid and aborts processing.

Once a suitable set of objects and triples has been produced, whether through combination, scavenging or back-off, the Linker must still decide which of the objects are to be displayed – the "topic" of the utterance. The topic-choice module for the Semantic Linker is fairly similar to the topic-choice module of the Frame Combiner reported on in [1], and so we do not go into much detail on it here. Basically, there are a number of heuristics, including whether the determiner of a nominal object is WH, whether the sort of the the nominal is a "priority" domain (in ATIS, GROUND-TRANSPORTATION is such a domain), and whether the nominal occurs only has the second argument of the triples in which it occurs (making it an unconstrained nominal). The important new feature of the Semantic Linker's topic choice module is its ability to make of use of links between a nominal object and a verb like "show" as evidence for topic choice.

6. RESULTS AND DISCUSSION

Results from the November 1992 DARPA evaluation[6] show that the Semantic Linker reduced DELPHI's Weighted Error rate on the NL-only portion of the test by 30% (from 32% to 22%). This was achieved mostly by dramaticaly lowering the No Answer rate (from 21% to 8%).

It should be noted that these results were achieved with an earlier version of the Semantic Linker than that reported here. In particular, this earlier version did not make use of empirically determined probabilities, but rather used a more ad hoc system of heuristically determined weights and features. Nevertheless, these preliminary results give us some confidence in our approach.

Several areas of future work are seen. One is the use of automatic training methods to determine feature weights. A corpus pairing sentences and sets of connecting links could be used in supervised training to adjust initial values of these weights up or down.

Another area, one in which we are already engaged, is using the Semantic Linker in ellipsis processing by treating the preceding utterance as a fragment-structure into which to link the present, elliptical one.

A third area of future work is the use of relational probabilities and search in the generation of fragments themselves. Currently, the fragment generator component is entirely separate from the rest of the Linker, which makes it difficult for combination search to recover from fragment generation. Instead of trying to combine fragments, the Linker could seek to combine the semantic objects internal to them, in a process where inter-object links found by the fragment generator would have a strong but not insurmountable advantages

A last area of future work is to more fully integrate the Semantic Linker into the regular parsing mechanism itself, and to investigate ways in which parsing can be viewed as similar to the linking process.

References

1. Stallard, D. and Bobrow, R.
Fragment Processing in the DELPHI System
Proceedings Speech and Natural Language Workshop February 1992
2. Seneff, Stephanie
A Relaxation Method for Understanding Spontaneous Speech Utterances
Proceedings Speech and Natural Language Workshop
February 1992
3. Linebarger, Marcia C., Norton, Lewis M., and Dahl, Deborah A.
A Portable Approach to Last Resort Parsing and Interpretation
(this volume)
4. Jackson, E., Appelt D., Bear J., Moore, R. and Podlozny, A.
A Template Matcher for Robust NL Interpretation
Proceedings Speech and Natural Language Workshop
February 1991
5. Bobrow, Robert
Statistical Agenda Parsing
Proceedings Speech and Natural Language Workshop
February 1991
6. Pallet, D., Fiscus, J., Fisher, W. and Garofolo, J.
Benchmark Tests for the Spoken Language Program
(this volume)