Charting the Depths of Robust Speech Parsing

W. Kasper[†], B. Kiefer[†], H.-U. Krieger[†], C. J. Rupp[‡], and K. L. Worm[‡]
[†]German Research Center for Artificial Intelligence (DFKI)
[‡]Computational Linguistics Department, Universität des Saarlandes
{kasper,kiefer,krieger}@dfki.de and {cj,worm}@coli.uni-sb.de

Abstract

We describe a novel method for coping with ungrammatical input based on the use of chart-like data structures, which permit anytime processing. Priority is given to deep syntactic analysis. Should this fail, the best partial analyses are selected, according to a shortest-paths algorithm, and assembled in a robust processing phase. The method has been applied in a speech translation project with large HPSG grammars.

1 Introduction

This paper describes a new method of dealing robustly with deficient speech or text input, which may be due to recognition errors, spontaneous speech phenomena, or ungrammatical constructions. Two key features of this approach are:

- the priority given to a deep and restrictive grammatical analysis,
- and the use of chart-like data structures at each level of processing.

The initial input is taken from a *Word Hypothesis Graph*, or WHG, (Oerder and Ney, 1993) from which the best ranked paths are successively selected until a result is found or a time limit proportional to the length of the utterance[1] is reached. Each path is parsed with an incremental chart parser that uses a Head-Driven Phrase Structure grammar (HPSG). The parser is adapted to input from WHGs and optimized to meet the needs of real-time speech processing. Since the goal of the parsing component is to process as many WHG paths as possible, in order to find a grammatical utterance and analyze it with highest accuracy, neither relaxation of the constraints imposed by the grammar nor repair rules are used at this stage. If the analysis of the current path is successful, the parsing process is complete. However, in most cases there is no spanning and syntactically correct analysis. So a sequence of partial analyses is determined by incrementally evaluating the passive edges in the parser's chart. These analyzed fragments are passed on to a robust semantic processing component for further treatment, while the next best WHG path is analyzed by the parser[2]. Robust semantic processing similarly builds up a chart-like data structure including analyzed fragments and the results of applying robustness rules at the semantic level. After the first path of the WHG has been (unsuccessfully) analyzed, processing in both the restrictive parser and the robustness component proceeds in parallel, with the aid of a parallel virtual machine, until one of the following conditions is fulfilled:

1. a spanning grammatical analysis is found,
2. all the WHG paths have been explored, or
3. the time limit is reached.

In the case of either of the latter two conditions, robust semantic processing is allowed a limited time to complete processing and then the best result or sequence of results is selected from its chart.

Our approach has been implemented in VERBMOBIL (Wahlster, 1993), a large scale research project in the area of spoken language

[1]This is currently up to four times real time.

[2]This means that the maximal sequential delay between parsing and robust semantics processing is the parse time for one path. Similarly, the limit on parsing time, essentially, applies to both components

translation. Its goal is to develop a system that translates negotiation dialogues between German, English, and Japanese speakers in face-to-face or video conferencing situations. This application highlights the basic problem associated with machine processing of spontaneous speech, namely that the input to the natural language processing component is perturbed by two influences:

1. Speakers make mistakes, correct themselves during speaking, produce false starts and use ungrammatical constructions.
2. The acoustic signal produced by a human speaker is mapped by a speech recognizer onto a written form; this mapping is rarely completely correct.

This introduces two levels of uncertainty into the processing of speech, which make the task of linguistically analyzing a spoken utterance in a speech processing system doubly hard. In addition, the dialogue context imposes strict time constraints, as the overall system must attempt to emulate real time performance.

The strategy we adopt responds to time constraints by universally incorporating an anytime property (Dean and Boddy, 1988) into the selection procedures. As will be seen, this property derives from the way in which intermediate results are stored and the selections which can be made from among these. However, the overriding priority of this same strategy is to maximize the chance that a truly grammatical path will be found and analyzed, if one exists in the WHG. This means that while we have implemented extensive mechanisms to achieve robustness, their design, and in particular the separation of processing into a restrictive parser and a robust postprocessor, are subservient to the cases where a fully grammatical analysis is possible, since these results are in any case better. These decisions may be in conflict with much of the literature on robust parsing (e.g., (Hindle, 1983; Hipp, 1993; Heinecke et al., 1998)), but the alternative of relaxing the parsing constraints would appear to be a dead end in the context of the VERBMOBIL architecture. In the first place, the chances of locating the best grammatical path in the lattice would be reduced, e.g., by the acceptance of a preceding ungrammatical one. Secondly, a more liberal parser would raise the spectre of an explosion of edges in the parser's chart, so that in fact less paths could be processed overall, regardless of their quality. Either of these conditions could prove fatal.

This paper focuses on the aspects of the VERBMOBIL analysis component which ensure that the most accurate results available are provided to the system as a whole. We first describe the basic inventory we need to explain our approach: the unification-based bottom-up chart parser, the HPSG grammar, and the interface terms which are exchanged between the parser and the robust semantic processing. After that, we come to the basic algorithm which determines best partial analyses. We also give an example of how the evaluation function on edges might look. In section 4, we focus on the robust semantic processing whose task is to store and combine the partial results, before choosing a final result out of a set of possible candidates. We end this paper by presenting empirical results on the usefulness of our approach.

2 Preliminaries

2.1 The Chart Parser

The parser used in the system is a bottom-up chart parser. Since the grammar is a pure unification-based grammar, there is no context-free backbone and the chart edges are labelled with typed feature structures. At the moment, there is no local ambiguity packing of chart edges. Therefore, the worst case complexity of parsing is potentially exponential, but since the parser employs a best-first strategy, exponential behavior is rarely found in practice.

The parser provides a flexible priority system for guiding the parsing process, using parsing tasks on an agenda. A parsing task represents the combination of a passive chart edge and an active chart edge or a rule. When such a combination succeeds, new tasks are generated and for each new task, a priority is assigned.

This priority system helps to obtain good partial results, even in cases where the search space cannot be fully explored due to parsing time restrictions. A higher time bound would allow either the processing of more WHG paths or a more elaborate analysis of the given input, both

of which may lead to better results. The decision when to switch to the next best path of a given WHG depends on the length of the input and on the time already used. After the parsing of one path is finished, the passive edges of the chart form a directed acyclic graph which is directly used as input to compute best partial analyses.

We note here that the parser processes the n-best paths of a WHG fully incrementally. I.e., when the analysis of a new input path begins, only those input items are added to the chart that have not been part of a previously treated path. Everything else that has been computed up to that point remains in the chart and can be used to process the new input without being recomputed.

2.2 The HPSG Grammars

The grammars for English, German, and Japanese follow the paradigm of HPSG (Pollard and Sag, 1994) which is the most advanced unification-based grammatical theory based on typed feature structures. The fundamental concept is that of a *sign*, a structure incorporating information from all levels of linguistic analysis, such as phonology, morphology, syntax, and semantics. This structure makes all information simultaneously available and provides declarative interfaces between these levels. The grammars use *Minimal Recursion Semantics* (Copestake et al., 1996) as the semantic representation formalism, allowing us to deal with ambiguity by underspecification.

To give an impression of the size of grammars, we present the numbers for the German grammar. It consists of 2,389 types, 76 rule schemata, 4,284 stems and an average of six entries per stem. Morphological information is computed online which further increases the lexical ambiguity.

2.3 Partial Analyses and the Syntax-Semantics Interface

Our architecture requires that the linguistic analysis module is capable of delivering not just analyses of complete utterances, but also of phrases and even of lexical items in the special interface format of VITs (VERBMOBIL *Interface Terms*) (Bos et al., 1998). There are three considerations which the interface has to take into account:

1. Only maximal projections, i.e., complete phrases, are candidates for robust processing. This qualifies, e.g., prepositional and noun phrases. On the other hand, this approach leaves gaps in the coverage of the input string as not every word needs to be dominated by a maximal projection. In particular, verbal projections below the sentential level usually are incomplete phrases. The use of intermediate, incomplete projections is avoided for several reasons:
   - intermediate projections are highly grammar and language specific and
   - there are too many of them.
2. Phrases must be distinguished from elliptical utterances. A major difference is that elliptical utterances express a speech act. E.g., a prepositional phrase can be a complete utterance expressing an answer to a question (*On Monday.*) or a question itself (*On Monday?*). If the phrase occurs in a sentence, it is not associated with a speech act of its own. This distinction is dealt with in the grammars by specifying special types for these complete utterances, phrases, and lexical items.
3. For robust processing, the interface must export a certain amount of information from syntax and morphology together with the semantics of the phrase. In addition, it is necessary to represent semantically empty parts of speech, e.g., separable verb prefixes in German.

3 Computing Best Partial Analyses

In contrast to a traditional parser which never comes up with an analysis for input not covered by the grammar, our approach focuses on *partial analyses* without giving up the correctness of the overall deep grammar. These partial analyses are combined in a later stage (see Section 4) to form total analyses. But what is a partial analysis? Obviously a derivation (sub)tree licensed by the grammar which covers a continuous part of the input (i.e., a passive parser edge). But not every passive edge is a good candidate, since otherwise we would end up with perhaps thousands of them. Our ap-

proach lies in between these two extremes: computing a connected sequence of *best* partial analyses which cover the whole input. The idea here is to view the set of passive edges of a parser as a directed graph which needs to be *evaluated* according to a user-defined (and therefore grammar and language specific) metric. Using this graph, we then compute the *shortest paths* w.r.t. the evaluation function, i.e., paths through this graph with minimum cost.

Since this graph is acyclic and topologically sorted (vertices are integers and edges always connect a vertex to a larger vertex), we have chosen the DAG-shortest-path algorithm (Cormen et al., 1990) which runs in $\Theta(V+E)$. This fast algorithm is a solution to the single-source shortest-paths problem. We modified and extended this algorithm to cope with the needs we encountered in speech parsing: (i) one can use several start and end vertices (e.g., in the case of n-best chains or WHGs); (ii) all best shortest paths are returned (i.e., we obtain a shortest-path subgraph); and (iii) evaluation and selection of the best edges is done incrementally as is the case for parsing the n-best chains (i.e., only new passive edges entered into the chart are evaluated and may be selected by our shortest-path algorithm).

We now sketch the basic algorithm. Let $G = \langle V, E \rangle$ denote the set of passive edges, $\mathcal{S}$ the set of start vertices, $\mathcal{E}$ the set of end vertices, and let n be the vertex with the highest number (remember, vertices are integers): $n = max(V)$. In the algorithm, we make use of two global vectors of length n which store information associated with each vertex: *dist* keeps track of the distance of a vertex to one of the start vertices (the so-called shortest-path estimate), whereas *pred* records the predecessors of a given vertex. *weight* defines the cost of an edge and is assigned its value during the evaluation stage of our algorithm according to the user-defined function *Estimate*. Finally, *Adj* consists of all vertices adjacent to a given vertex (we use an adjacency-list representation).

Clearly, before computing the shortest path, the distance of a vertex to one of the start vertices is infinity, except for the start vertices, and there is of course no shortest path subgraph ($pred(v) \leftarrow \emptyset$).

```
Initialise-Single-Source(G, S) :⟺
  global dist, pred;
  for each v ∈ V(G) do
    dist(v) ← ∞;
    pred(v) ← ∅
    od;
  for each s ∈ S do
    dist(s) ← 0
    od.
```

After initialization, we perform evaluation and relaxation on every passive edge, taken in topologically sorted order. Relaxing an edge $\langle u, v \rangle$ means checking whether we can improve the shortest path(s) to v via u. There are two cases to consider: either we overwrite the shortest-path estimate for v since the new one is better (and so have a new predecessor for v, viz., u), or the shortest-path estimate is as good as the old one, hence we have to add v to the predecessors of v. In case the shortest-path estimate is worse, there is clearly nothing to do.

```
Relax(u, v) :⟺
  global dist, pred;
  if dist(v) > dist(u) + weight(u, v)
   then do
     dist(v) ← dist(u) + weight(u, v);
     pred(v) ← {u}
     od
   else do
     when dist(v) = dist(u) + weight(u, v) do
       pred(v) ← pred(v) ∪ {u}
       od
     od
   fi.
```

The shortest paths are then determined by estimating and relaxing edges, beginning with the start vertices $\mathcal{S}$. The shortest path subgraph is stored in *pred* and can be extracted by walking from the end vertices $\mathcal{E}$ 'back' to the start vertices.

```
DAG-Shortest-Paths(G, S, E) :⟺
  global pred;
  Initialise-Single-Source(G, S);
  for each u ∈ V(G) \ E taken in topologically
  sorted order do
    for each v ∈ Adj(u) do
      weight(u, v) ← Estimate(u, v);
      Relax(u, v)
    od
  od;
  return pred.
```

After we have determined the shortest-path subgraph, the feature structures associated with these edges are selected and transformed to the corresponding VITs which are then sent to the robust semantic processing component.

This approach has an important property: even if certain parts of the input have not undergone at least one rule application, there are still lexical edges which help to form a best path through the passive edges. Hence, this approach shows anytime behavior which is a necessary requirement in time-critical (speech) applications: even if the parser is interrupted at a certain point, we can always return a shortest path up to that moment through our chart.

Let us now give an example to see what the evaluation function on edges (i.e., derivation trees) might look like[3]:

- n-ary trees ($n > 1$) with utterance status (e.g., NPs, PPs): value 1
- lexical items: value 2
- otherwise: value ∞

If available, other properties, such as prosodic information or probabilistic scores can also be utilized in the evaluation function to determine the best edges.

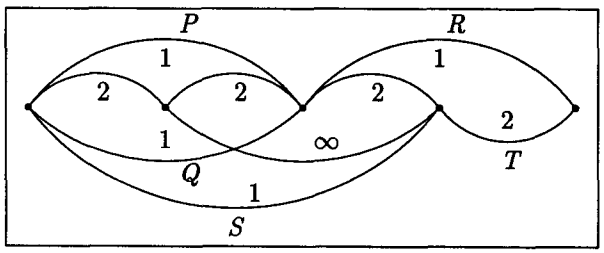

Figure 1: Computing best partial analyses. Note that the paths PR and QR are chosen, but not ST, although S is the longest edge. By using uniform costs, all three paths would be selected.

Depending on the evaluation, our method does not necessarily favor paths with longest edges as the example in Figure 1 shows — the above strategy instead prefers paths containing no lexical edges (where this is possible) and there might be several such paths having the same cost. Longest (sub)paths, however, can be obtained by employing an exponential functions during the evaluation of an edge $e \in E$: $Estimate(e) = -(max(\mathcal{E}) - min(\mathcal{S}))^{length(e)}$.

[3]This is a slightly simplified form of the evaluation that is actually used for the German grammar.

4 Robust Semantic Processing

The second phase of processing, after producing a set of partial analyses, consists of assembling and combining the fragments, where possible. We call this *robust semantic processing* (Worm and Rupp, 1998), since the structures being dealt with are semantic representations (VITs) and the rules applied refer primarily to the semantic content of fragments, though they also consider syntactic and prosodic information, e.g., about irregular boundaries.

This phase falls into three tasks:

1. *storing* the partial analyses from the parser,
2. *combining* them on the basis of a set of rules, and
3. *selecting* a result.

For storing of partial results, both delivered from the parser or constructed later, we make use of a chart-like data structure we call *VIT hypothesis graph* (VHG), since it bears a resemblance to the WHG which is input to the parser. It is organized according to WHG vertices. We give an example in Figure 2, which will be explained in 4.1.

Combination of partial results takes place using a set of rules which describe how fragmentary analyses can be combined. There are language-independent rules, e.g., describing the combination of a semantic functor with a possible argument, and language specific ones, such as those for dealing with self-corrections in German. Each operation carried out delivers a confidence value which influences the score assigned to an edge.

The overall mechanism behind the robust semantic processing resembles that of a chart parser. It runs in parallel with the HPSG parser; each time the parser delivers partial results, they are handed over and processed, while the parser may continue to look for a better path in the WHG. The processing strategy is

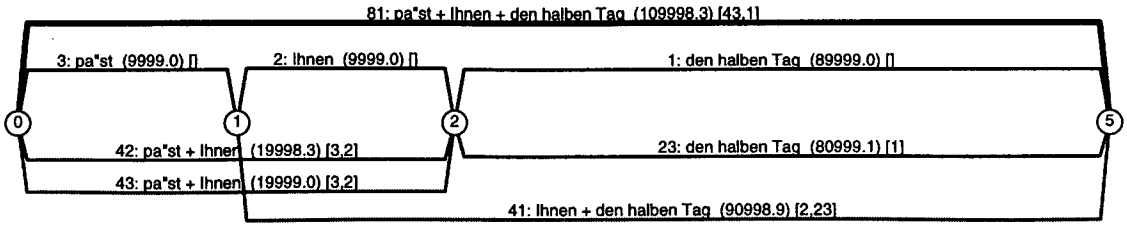

Figure 2: The VHG for the first example. Only three VITs are delivered by the parser (the shortest path), although the number of passive edges is 217.

agenda-based, giving priority to new parser results.

Selection of a result means that the best edge covering the whole input, or if that has not been achieved, an optimal sequence of edges has to be selected. We use a simple graph search algorithm which finds the path with the highest sum of individual scores.

Note that the robust semantic processing has the anytime property as well: as soon as the first partial result has been entered into the chart, a result can be delivered on demand.

4.1 An Example

Consider the utterance (1) where the case of the NP *den halben Tag* ('half the day') is accusative and thus does not match the subcategorization requirements of the verb *passen* ('suit') which would require nominative.

(1) Paßt Ihnen den halben Tag?
'Does half the day suit you?'

According to the grammar, this string is ill-formed, thus no complete analysis can be achieved. However, the parser delivers fragments for *paßt*, *Ihnen*, and *den halben Tag*.

(2)
```
verb_arg_r ::
[[type(V1,verbal),missing_arg(V1)],
 [type(V2,term),possible_arg(V2,V1)]]
--->
[apply_fun(V1,V2,V3),
 assign_mood(V3,V4)] & V4.
```

When these results are stored, the rule in (2) will combine the verb with its first argument, *Ihnen*. Each rule consists of three parts: mnemonic rule name, tests on a sequence of input VITs and the operations performed to construct the ouput VIT. The first separator is ::, the second --->. A further application of the same rule accounts for the second argument, *den halben Tag*. However, the confidence value for the second combination will reflect the violation of the case requirement. The resulting edge spans the whole input and is selected as output. The corresponding VHG is shown in Figure 2.

4.2 Bridging

Not all cases can be handled as simply. Often, there are partial results in the input which cannot be integrated into a spanning result. In these cases, a mechanism called *bridging* is applied. Consider the self-correction in (3).

(3) Ich treffe ... habe einen Termin am Montag.
'I (will) meet ... have an appointment on Monday.'

Again, the parser will only find partial results. Combinations of *ich* with *treffe* lead nowhere; the combination of the second verb with the NP does not lead to a complete analysis either (cf. Figure 3). Note that if a nominal argument can bind several argument roles, for each such reading there is a passive edge in the VHG. Its score reflects to what degree the selectional requirements of the verb, in terms of the required case and sortal restrictions, have been met.

If no spanning result exists when all rules have been applied, the bridging mechanism produces new active edges which extend edges already present. Here, it extends the active edge aiming to combine *ich* with a verbal functor to end after *treffe*, thus allowing for a combination with the VP already built, *habe einen Termin*

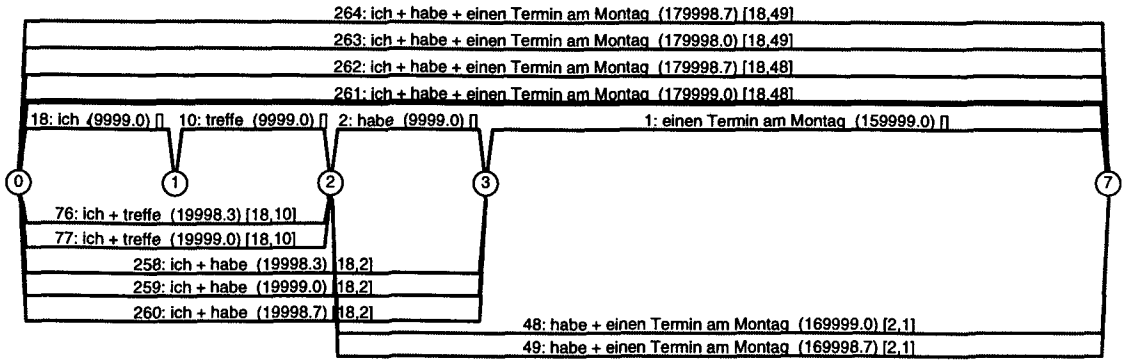

Figure 3: The VHG for the second example.

am Montag. Extending the active edges from left to right corresponds to the linear nature of self-corrections, in which material to the right replaces some to the left.

4.3 Scoring and Result Selection

The scoring function for edges takes into account their length, the coverage of the edge, the number of component edges it consists of, and the confidence value for the operation which created it. It has to satisfy the following property, which is illustrated in Figure 4: If there are two edges which together span an interval (edges *a* and *b*) and another edge which has been built from them (edge *c*), the latter should get a better score than the sequence of the original two edges. If there is another edge from the parser which again spans the complete interval (edge *d*), it should get a better score than the edge built from the two components.

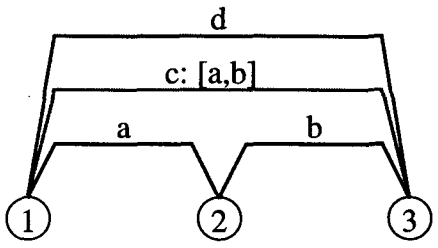

Figure 4: Requirements for the scoring function.

The selection is done in two different ways. If there is more than one spanning result, the scores of the spanning results are weighted according to a statistical model describing sequence probabilities based on semantic predicates (Ruland et al., 1998) and the best is selected. Otherwise, the best sequence, i.e., the one with the highest score, is chosen in square time, using a standard graph search algorithm.

5 Empirical Results

For an intermediate evaluation of the robust semantic processing phase, we ran our system consisting of HPSG parser and robust semantic processing on a dialogue from the VERBMOBIL corpus of spontaneous appointment negotiation dialogues, producing WHGs from the original recorded audio data. The dialogue consists of 90 turns. These 90 turns were split into 130 segments according to pauses by the speech recognizer. The segments received 213 segment analyses, i.e., there are 1.6 analyses per segment on average. 172 (80.8%) of these were generated by the parser and 41 (19.2%) were assembled from parser results by robust semantic processing. Of these 41 results, 34 (83%) were sensibly improved, while 7 (17%) did not represent a real improvement.

This evaluation is local in the sense that we only consider the input-output behaviour of robust semantic processing. We do this in order to exclude the effects of insufficiencies introduced by other modules in the system, since they would distort the picture. For this same reason, the criterion we apply is whether the result delivered is a sensible combination of the frag-

ments received, without reference to the original utterance or the translation produced. However, in the long run we plan to compare the complete system's behaviour with and without the robust processing strategy.

6 Conclusion

The approach to the robust analysis of spoken language input, that we have described above, exhibits three crucial properties.

1. The restrictive parser is given the maximum opportunity of finding a correct analysis for a grammatical sequence of word hypotheses, where this exists.
2. The robustness component assembles partial analyses as a fallback, if no grammatical sequence can be found.
3. Almost arbitrary time constraints can be supported. Though, obviously, more processing time would usually improve the results.

The latter property depends directly on the chart-like data structures used at each level of processing. Whether it be the input WHG, VHG for robust processing or, most significantly, the parser's chart; each is formally a directed acyclic graph and each permits a selection of the best intermediate result at, virtually, any stage in processing, for a given evaluation function.

The relatively efficient processing of WHG input achieved by parsing and robustness components working in parallel depends quite heavily on the successive processing of ranked WHG paths, effectively as alternative input strings.

Acknowledgments

We would like to thank the anonymous ACL reviewers for their detailed comments. This research was supported by the German Federal Ministry for Education and Research under grants nos. 01 IV 701 R4 and 01 IV 701 V0.

References

Johan Bos, C.J. Rupp, Bianka Buschbeck-Wolf, and Michael Dorna. 1998. Managing information at linguistic interfaces. In *Proc. of the 17th COLING/36th ACL*, pages 160–166, Montréal, Canada.

Ann Copestake, Dan Flickinger, and Ivan A. Sag. 1996. Minimal recursion semantics. an introduction. Ms, Stanford.

Thomas H. Cormen, Charles E. Leiserson, and Ronald L. Rivest. 1990. *Introduction to Algorithms.* MIT Press, Cambridge, MA.

Thomas Dean and Mark Boddy. 1988. An analysis of time-dependent planning. In *Proceedings of the 7th National Conference on Artificial Intelligence, AAAI-88*, pages 49–54.

Johannes Heinecke, Jürgen Kunze, Wolfgang Menzel, and Ingo Schröder. 1998. Eliminative parsing with graded constraints. In *Proc. of the 17th COLING/36th ACL*, pages 526–530, Montréal, Canada.

Donald Hindle. 1983. Deterministic parsing of syntactic non-fluencies. In *Proc. of the 21 th ACL*, pages 123–128, Cambridge, MA.

Dwayne Richard Hipp. 1993. *Design and Development of Spoken Natural-Language Dialog Parsing Systems.* Ph.D. thesis, Department of Computer Science, Duke University, Durham, NC.

Martin Oerder and Hermann Ney. 1993. Word graphs: An efficient interface between continuous-speech recognition and language understanding. In *Proc. Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, Minneapolis, MN. IEEE Signal Processing Society.

Carl Pollard and Ivan A. Sag. 1994. *Head-Driven Phrase Structure Grammar.* University of Chicago Press, Chicago.

Tobias Ruland, C. J. Rupp, Jörg Spilker, Hans Weber, and Karsten L. Worm. 1998. Making the most of multiplicity: A multi-parser multi-strategy architecture for the robust processing of spoken language. In *Proc. of the 1998 International Conference on Spoken Language Processing (ICSLP 98)*, Sydney, Australia.

Wolfgang Wahlster. 1993. VERBMOBIL — translation of face-to-face dialogs. In *Proc. MT Summit IV*, pages 127–135, Kobe, Japan, July.

Karsten L. Worm and C. J. Rupp. 1998. Towards robust understanding of speech by combination of partial analyses. In *Proc. of the 13 th ECAI*, pages 190–194, Brighton, UK.