Partial Parse Selection for Robust Deep Processing

Yi Zhang† and **Valia Kordoni**† and **Erin Fitzgerald**‡
† Dept of Computational Linguistics, Saarland University and DFKI GmbH, Germany
‡ Center for Language & Speech Processing,
Dept of Electrical & Computer Engineering, Johns Hopkins University, USA
{yzhang,kordoni}@coli.uni-sb.de
erin@clsp.jhu.edu

Abstract

This paper presents an approach to partial parse selection for robust deep processing. The work is based on a bottom-up chart parser for HPSG parsing. Following the definition of partial parses in (Kasper et al., 1999), different partial parse selection methods are presented and evaluated on the basis of multiple metrics, from both the syntactic and semantic viewpoints. The application of the partial parsing in spontaneous speech texts processing shows promising competence of the method.

1 Introduction

Linguistically deep processing is of high theoretical and application interest because of its ability to deliver fine-grained accurate analyses of natural language sentences. Unlike shallow methods which usually return analyses for any input, deep processing methods with precision grammars normally make a clear grammaticality judgment on inputs, therefore avoiding the generation of erroneous analyses for less well-formed inputs. This is a desirable feature, for it allows for a more accurate modeling of language itself.

However, this feature largely limits the robustness of deep processing, for when a sentence is judged to be ungrammatical, normally no analysis is generated. When faced with the noisy inputs in real applications (e.g., input errors introduced by speech recognizers or other pre-processors, mildly ungrammatical sentences with fragmental utterances, self-editing chunks or filler words in spoken texts, and so forth), lack of robustness means poor coverage, and makes deep processing less competitive as compared to shallow methods.

Take the English Resource Grammar (ERG; Flickinger (2000)), a large-scale accurate HPSG for English, for example. (Baldwin et al., 2004) reported coverage of 57% of the strings with full lexical span from the British National Corpus (BNC). Although recent extensions to the grammar and lexicon have improved the coverage significantly, full coverage over unseen texts by the grammar is still not anywhere in sight.

Other domains are even more likely to not fit into ERG's universe, such as transcripts of spontaneously produced speech where speaker errors and disfluencies are common. Using a recent version of the ERG, we are not able to parse 22.6% of a random sample of 500 utterances of conversational telephone speech data. 76.1% of the unparsed data was independently found to contain speaker errors and disfluencies, and the remaining data either contained filled pauses or other structures unaccounted for in the grammar. Correctly recognizing and interpreting the substrings in the utterance which have coherent deep syntax is useful both for semantic analysis and as building blocks for attempts to reconstruct the disfluent spontaneously produced utterances into well-formed sentences.

For these reasons, it is preferable to exploit the intermediate syntactic and semantic analysis even if the full analysis is not available. Various efforts have been made on the partiality of language processing. In bottom-up chart parsing, the passive parser edges licensed by the grammar can be taken as partial analyses. However, as pointed out in (Kasper et al., 1999), not all passive edges are good candidates, as not all of them provide useful syntactic/semantic information. Moreover, the huge amount of passive edges suggests the need for a technique of selecting an optimal subset of them. During recent development in statistical parse disambiguation, the use of log-linear models has been pretty much standardized. However, it remains to be explored whether the techniques can be adapted for partial parse selection.

In this paper, we adopt the same definition for partial parse as in (Kasper et al., 1999) and define the task of partial parse selection. Several dif-

Proceedings of the 5th Workshop on Important Unresolved Matters, pages 128–135,
Ann Arbor, June 2005.

ferent partial parse selection models are presented and implemented for an efficient HPSG parser – PET (Callmeier, 2001).

One of the main difficulties in the research of partial analyses is the lack of good evaluation measurements. Pure syntactic comparisons for parser evaluation are not good as they are very much specific to the annotation guidelines. Also, the deep grammars we are working with are not automatically extracted from annotated corpora. Therefore, unless there are partial treebanks built specifically for the deep grammars, there is simply no 'gold' standard for non-golden partial analyses.

Instead, in this paper, we evaluate the partial analyses results on the basis of multiple metrics, from both the syntactic and semantic point of views. Empirical evaluation has been done with the ERG on a small set of texts from the Wall Street Journal Section 22 of the Penn Treebank (Marcus et al., 1993). A pilot study of applying partial parsing in spontaneous speech text processing is also carried out.

The remainder of the paper is organized as follow. Section 2 provides background knowledge about partial analysis. Section 3 presents various partial parse selection models. Section 4 describes the evaluation setup and results. Section 5 concludes the paper.

2 Partial Parsing

2.1 HPSG Parsing

Our work on partial parsing is done with the DELPH-IN HPSG grammars. Many of these grammars can be used for both parsing and generation. In this paper, we only focus on the parsing task. For efficient parsing, we use PET.[1] The parsing module in PET is essentially a bottom-up chart parser. The parsing process is guided by the 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 the combination succeeds, new tasks are generated and put on to the agenda. The parser terminates either when the task agenda is empty or when a specific number of full analyses has been found (only in the no-packing best-first mode).

HPSG grammars use typed feature structures (TFSes) as their background formalism. The TFSes represent various linguistic objects with a set of features (attribute value pairs) and a type inheritance system. Therefore, each passive edge on the parsing chart corresponds to a TFS. A relatively small set of highly generalized rules are used to check the compatibility among smaller TFSes and build up larger ones.

2.2 Partial Parses

Based on the bottom-up chart parsing, we use the term *Partial Parse* to describe a set of intermediate passive parsing edges whose spans (beginning and end positions) are non-overlapping between each other, and together they cover the entire input sequence (i.e., no skipped input tokens).

In a graph view, the intermediate results of a chart parser can be described as a directed graph, where all positions between input tokens/words are vertices, and all the passive edges derived during parsing are the directed graph arcs. Obviously such a graph is acyclic and therefore topologically sorted. A partial parse is then a path from the source vertex (the beginning position of the input) to the terminal vertex (the end position of the input).

Suppose in chart parsing, we derived the intermediate results as in Figure 1. There are in total 4 possible partial parses: $\{a, b, c, d\}$, $\{a, b, f\}$, $\{a, e, d\}$ and $\{a, g\}$.

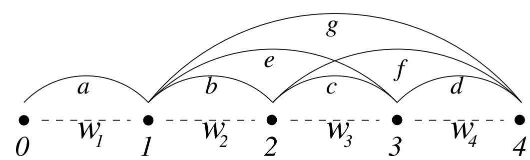

Figure 1: Graph representation of intermediate chart parsing results

Note that each passive edge is a sub-structure licensed by the grammar. A derivation tree or TFS can be reconstructed for it if required. This definition of *partial parse* is effectively the same to the view of partial analyses in (Kasper et al., 1999).

2.3 Local Ambiguity Packing

There is one more complication concerning the partial parses when the local ambiguity packing is used in the parser.

Due to the inherent ambiguity of natural language, the same sequence of input may be analyzed as the same linguistic object in different ways. Such intermediate analyses must be recorded during the processing and recovered in later stages.

[1] LKB (Copestake, 2002) has a similar chart-based parser, being less efficient mainly due to its implementation in Lisp rather than C/C++.

Without any efficient processing technique, parsing becomes computationally intractable with the combinatory explosion of such local ambiguities. In PET, the subsumption-based ambiguity packing algorithm proposed in (Oepen and Carroll, 2000) is used. This separates the parsing into two phases: forest creation phase and read-out/unpacking phase.

In relation to the work on partial parsing in this paper, the local ambiguity packing poses an efficiency and accuracy challenge, as not all the intermediate parsing results are directly available as passive edges on the chart. Without unpacking the ambiguity readings, interesting partial analyses might be lost.[2] But exhaustively unpacking all the readings will pay back the efficiency gain by ambiguity packing, and eventually lead to computational intractable results.

To efficiently recover the ambiguous readings from packed representations, the selective unpacking algorithm has been recently implemented as an extension to the algorithm described in (Carroll and Oepen, 2005). It is able to recover the top-n best readings of a given passive parser edge based on the score assigned by a maximum entropy parse ranking model. This neat feature largely facilitates the efficient searching for best partial parses described in later sections.

3 Partial Parse Selection

A partial parse is a set of partial analyses licensed by the grammar which cover the entire input without overlapping. As shown in the previous section, there are usually more than one possible partial parses for a given input. For deep linguistic processing, a high level of local ambiguity means there are even more partial parses due to the combinatory explosion. However, not all the possible partial parses are equally good. Some partial parses partition the input into fragments that do not correspond to linguistic constituents. Even if the bracketing is correct, the different edges with the same span represent significantly different linguistic objects, and their substructures can be completely different, as well. All these indicate the need for methods that can appropriately select the best partial parses from all the possible ones.

In this section, we review some of the previous approaches to partial parse selection, as well as new partial parse ranking models.

3.1 Longest Edge

One of the simplest and most commonly used criterion in selecting the best partial parse is to prefer the partial parses which contain an edge that covers the largest fragment of the input. For example, under such a criterion, the best partial parse in Figure 1 will be $\{a, g\}$, since edge g has the largest span. The logic behind this criterion is that such largest fragments should preserve the most interesting linguistic analysis of the input. As an added incentive, finding the longest edge does not involve much search.

The limitations of such an approach are obvious. There is no guarantee that the longest edge will be significantly better than shorter edges, or that it will even correspond to a valid constituent. Moreover, when there are multiple edges with the same length (which is often the case in parsing), the criterion does not suffice for the choice of the best partial parse.

3.2 Shortest Path

(Kasper et al., 1999) proposed an alternative solution to the problem. If the preference of each edge as a part of the partial parse can be quantitatively decided as a weight of the edge (with smaller weights assigned to better candidates), then the problem of finding the best partial parse is to find the shortest path from the start vertex to the end vertex. Since the graph is completely connected (by the lexical edges spanning all the input tokens) and topologically sorted, such a path always exists. The discovery of such a path can be done in linear time ($O(|V| + |E|)$) with the DAG-shortest-path algorithm (Cormen et al., 1990). Though not explicitly pointed out by (Kasper et al., 1999), such an algorithm allows the weights of the edges to be of any real value (no assumption of positive weights) as long as the graph is a Directed Acyclic Graph (DAG).

(Kasper et al., 1999) did point out that the weights of the edges can be assigned by an estimation function. For example, the implementation of the algorithm in PET preferred phrasal edges over lexical edges. Other types of edges are not allowed in the partial parse. Suppose that we assign weight 1 to phrasal edges, 2 to lexical edges, and $\inf$ to all other edges. Then for the graph in 2, the best partial parses are $\{e, g\}$ and $\{f, g\}$, both of which have

[2] More informative analyses are subsumed by less informative ones. In subsumption-based packing, such analyses are packed and are not directly accessible.

the path length of 2. It should be noted that such an approach does not always favor the paths with the longest edges (i.e., path $\{h, d\}$ is not preferred in the given example).

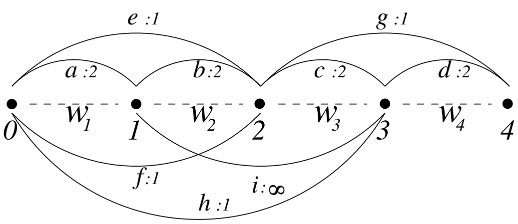

Figure 2: Shortest path partial parses with heuristically assigned edge weights

However, (Kasper et al., 1999) did not provide any sophisticated estimation functions based on the shortest path approach. Using the heuristic weight described above, usually thousands of different paths are found with the same weight. (Kasper et al., 1999) rely on another scoring function in order to re-rank the partial parses. Although different requirements for the scoring function are discussed, no further details have been defined.

It should be noted that different variations of the shortest path approach are widely in use in many robust deep parsing systems. For instance, (Riezler et al., 2002) uses the *fewest chunk* method to choose the best fragment analyses for sentences without full analysis. The well-formed chunks are preferred over token chunks. With this partial parse selection method, the grammar achieves 100% coverage on unseen data. A similar approach is also used in (van Noord et al., 1999).

3.3 Alternative Estimation Functions

Generally speaking, the weights of the edges in the shortest path approach represent the quality of the local analyses and their likelihood of appearing in the analysis of the entire input.

This is an interesting parallel to the parse selection models for the full analyses, where a goodness score is usually assigned to the full analysis. For example, the parse disambiguation model described in (Toutanova et al., 2002) uses a maximum entropy approach to model the conditional probability of a parse for a given input sequence $P(t|w)$. A similar approach has also been reported in (Johnson et al., 1999; Riezler et al., 2002; Malouf and van Noord, 2004).

For a given partial parse $\Phi = \{t_1, \ldots, t_k\}$, $\Omega = \{w_1, \ldots, w_k\}$ is a segmentation of the input sequence so that each local analysis $t_i \in \Phi$ corresponds to a substring $w_i \in \Omega$ of the input sequence w. Therefore, the probability of the partial parse Φ given an input sequence w is:

$$P(\Phi|w) = P(\Omega|w) \cdot P(\Phi|\Omega) \quad (1)$$

With the bold assumption that $P(t_i|w_i)$ are mutually independent for different i, we can derive:

$$P(\Phi|w) \approx P(\Omega|w) \cdot \prod_{i=1}^{k} P(t_i|w_i) \quad (2)$$

Therefore, the log-probability will be

$$\log P(\Phi|w) \approx \log P(\Omega|w) + \sum_{i=1}^{k} \log P(t_i|w_i) \quad (3)$$

Equation 3 indicates that the log-probability of a partial parse for a given input is the sum of the log-probability of local analyses for the sub-strings, with an additional component $-\log P(\Omega|w)$ representing the conditional log-probability of the segmentation. If we use $-\log P(t_i|w_i)$ as the weight for each local analysis, then the DAG shortest path algorithm will quickly find the partial parse that maximizes $\log P(\Phi|w) - \log P(\Omega|w)$.

The probability $P(t_i|w_i)$ can be modeled in a similar way to the maximum entropy based full parse selection models:

$$P(t_i|w_i) = \frac{\exp \sum_{j=1}^{n} \lambda_j f_j(t_i, w_i)}{\sum_{t' \in T} \exp \sum_{j=1}^{n} \lambda_j f_j(t', w_i)} \quad (4)$$

where T is the set of all possible structures that can be assigned to w_i, $f_1 \ldots f_n$ are the features and $\lambda_1 \ldots \lambda_n$ are the parameters. The parameters can be efficiently estimated from a treebank, as shown by (Malouf, 2002). The only difference from the full parse selection model is that here intermediate results are used to generate events for training the model (i.e. the intermediate nodes are used as positive events if it occurs on one of the active tree, or as negative events if not). Since there is a huge number of intermediate results availalbe, we only randomly select a part of them as training data. This is essentially similar to the approach in (Osborne, 2000), where there is an infeasibly large number of training events, only part of which is used in the estimation step. The exact features used in the log-linear model can significantly influence the disambiguation accuracy. In this experiment we used the same features as those used in the PCFG-S model in (Toutanova et al., 2002) (i.e., depth-1 derivation trees).

The estimation of $P(\Omega|w)$ is more difficult. In a sense it is similar to a segmentation or chunking model, where the task is to segment the input into fragments. However, it is difficult to collect training data to directly train such a model for the deep grammar we have. Here we take a simple rough estimation:

$$\hat{P}(\Omega|w) = \frac{|Y(\Omega)|}{|Z(w)|} \tag{5}$$

where $Y(\Omega)$ is the set of all partial parses that have the segmentation Ω; $Z(w)$ is the set of all partial parses for the input w.

Unfortunately, the shortest path algorithm is not able to directly find the maximized $P(\Phi|w)$. Fully searching all the paths is not practical, since there are usually tens of thousands of passive edges. In order to achieve a balance between accuracy and efficiency, two different approximation approaches are taken.

One way is to assume that the component $\log P(\Omega|w)$ in Equation 3 has less significant effect on the quality of the partial parse. If this is valid, then we can simply use $-\log P(t_i|w_i)$ as edge weights, and use the shortest path algorithm to obtain the best Φ. This will be referred to as *model I*.

An alternative way is to first retrieve several "good" Ω with relatively high $P(\Omega|w)$, and then select the best edges t_i that maximize $P(t_i|w_i)$ for each w_i in Ω. We call this approach the *model II*.

How well these strategies work will be evaluated in Section 4. Other strategies or more sophisticated searching algorithms (e.g., genetic algorithm) can also be used, but we will leave that to future research.

3.4 Partial Semantic Construction

For each local analysis on the partial parse derived in the above steps, a semantic fragment can be derived. The HPSG grammars we use take a compositional approach to semantic construction. Minimal Recursion Semantics (MRS; Copestake et al. (2006)) is used for semantic representation. MRS can be easily converted to (Robust) MRS (RMRS; Copestake (2006)), which allows further underspecification, and can be used for integration of deep and/or shallow processing tools.

For robust deep processing, the ability to generate partial semantics is very important. Moreover, it also provides us with a way to evaluate the partial parses which is more or less independent from the syntactic analysis.

4 Evaluation

The evaluation of partial parses is not as easy as the evaluation of full parses. For full parsers, there are generally two ways of evaluation. For parsers that are trained on a treebank using an automatically extracted grammar, an unseen set of manually annotated data is used as the test set. The parser output on the test set is compared to the gold standard annotation, either with the widely used *PARSEVAL* measurement, or with more annotation-neutral dependency relations. For parsers based on manually compiled grammars, more human judgment is involved in the evaluation. With the evolution of the grammar, the treebank as the output from the grammar changes over time (Oepen et al., 2002). The grammar writer inspects the parses generated by the grammar and either "accepts" or "rejects" the analysis.

In partial parsing for manually compiled grammars, the criterion for acceptable analyses is less evident. Most current treebanking tools are not designed for annotating partial analyses. Large-scale manually annotated treebanks do have the annotation for sentences that deep grammars are not able to fully analyze. And the annotation difference in other language resources makes the comparison less straightforward. More complication is involved with the platform and resources used in our experiment. Since the DELPH-IN grammars (ERG, JaCY, GG) use MRS for semantics representation, there is no reliable way of evaluating the output with traditional metrics, i.e., dependency relations.

In this paper, we use both manual and automatic evaluation methods on the partial parsing results. Different processing resources are used to help the evaluation from the syntactic, as well as the semantic point of view.

4.1 Syntactic Evaluation

In order to evaluate the quality of the syntactic structures of the partial parses, we implemented the partial parse models described in the previous section in the PET parser. The Nov-06 version of the ERG is used for the experiment. As test set, we used a subset of sentences from the Wall Street Journal Section 22 from the Penn Treebank. The subset contains 143 sentences which do not receive any full analysis licensed by the grammar, and do not contain lexical gaps (input tokens for which the grammar cannot create any lexical edge). The average sentence length is 24 words.

Due to the inconsistency of the tokenisation, bracketing and branching between the Penn Treebank annotation and the handling in ERG, we manually checked the partial parse derivation trees. Each output is marked as one of the three cases: **GBL** if both the bracketing and the labeling of the partial parse derivation trees are good (with no more than two brackets crossing or four false labelings); **GB** if the bracketings of the derivation trees are good (with no more than two brackets crossing), but the labeling is bad (with more than four false labelings); or **E** if otherwise.

The manual evaluation results are listed in Table 1. The test set is processed with two models presented in Section 3.3 (**M-I** for *model I*, **M-II** for *model II*). For comparison, we also evaluate for the approach using the shortest path with heuristic weights (denoted by **SP**). In case there are more than one path found with the same weight, only the first one is recorded and evaluated.

	GBL		GB		E	
	#	%	#	%	#	%
SP	55	38.5%	64	44.8%	24	16.8%
M-I	61	42.7%	46	32.2%	36	25.2%
M-II	74	51.7%	50	35.0%	19	13.3%

Table 1: Syntactic Evaluation Results

The results show that the naïve shortest path approach based on the heuristic weights works pretty well at predicting the bracketing (with 83.3% of the partial parses having less than two brackets crossing). But, when the labeling is also evaluated it is worse than *model I*, and even more significantly outperformed by *model II*.

4.2 Semantic Evaluation

Evaluation of the syntactic structure only reflects the partial parse quality from some aspects. In order to get a more thorough comparison between different selection models, we look at the semantic output generated from the partial parses.

The same set of 143 sentences from the Wall Street Journal Section 22 of the Penn Treebank is used. The RMRS semantic representations are generated from the partial parses with different selection models. To compare with, we used RASP 2 (Briscoe et al., 2006), a domain-independent robust parsing system for English. According to (Briscoe and Carroll, 2006), the parser achieves fairly good accuracy around 80%. The reasons why we choose RASP for the evaluation are: i) RASP has reasonable coverage and accuracy; ii) its output can be converted into RMRS representation with the LKB system. Since there is no large scale (R)MRS treebank with sentences not covered by the DELPH-IN precision grammars, we hope to use the RASP's RMRS output as a standalone annotation to help the evaluation of the different partial parse selection models.

To compare the RMRS from the RASP and the partial parse selection models, we used the similarity measurement proposed in (Dridan and Bond, 2006). The comparison outputs a distance value between two different RMRSes. We normalized the distance value to be between 0 and 1. For each selection model, the average RMRS distance from the RASP output is listed in Table 2.

	RMRS Dist.(ϕ)
SP	0.674
M-I	0.330
M-II	0.296

Table 2: RMRS distance to RASP outputs

Again, we see that the outputs of *model II* achieve the highest similarity when compared with the RASP output. With some manual validation, we do confirm that the different similarity does imply a significant difference in the quality of the output RMRS. The shortest path with heuristic weights yielded very poor semantic similarity. The main reason is that not every edge with the same span generates the same semantics. Therefore, although the SP receives reasonable bracketing accuracy, it has less idea of the goodness of different edges with the same span. By incorporating $P(t_i|w_i)$ in the scoring model, the model I and II can produce RMRSes with much higher quality.

4.3 Evaluating partial parses on spontaneous speech text

The above evaluation shows in a comparative way that *model II* outperforms other selection models from both syntactic and semantic points of view. In order to show its competence in real applications, we applied the best performing *model II* on spontaneous speech transcripts, which have a high level of informality and irregularity not available in newspaper texts such as the Wall Street Journal.

To evaluate the accuracy and potential interpretational value of partial parsing on spontaneous speech transcripts, we considered a 100-sentence random sample of the Fisher Conversational Telephone Speech 2004 development subcorpus (Cieri et al., 2004), used in the fall 2004 NIST Rich Transcription task.

Of these 100 sentences, six utterances received neither full nor partial parses due to lexical gaps created by words not found in the grammar's lexicon.[3] 75 utterances produced full HPSG parses. For the remaining 19 utterances, the one best partial parse is found for each using *model II*.

According to manual evaluation of the output, semantically and syntactically cohesive partial analyses were successfully assigned to 9 of the 19 partially parsed utterances. 3 of the 19 received incomplete semantics. The remaining 7 were judged to be poor due to false segmentation, the syntax and semantics within those parsed fragments, or both. In one instance, the interpretation was plausible but viewed as far less likely by the evaluator than the preferable interpretation ("... [i think you know it it 's] [court]"[4]). It is likely that n-best partial parsing could help us in most cases. This would only require a straightforward extension of the current partial parsing models.

Current partial parsing models do not use any confidence thresholds. Therefore, any input will receive some full or partial analysis (ignoring the case of unknown words), together with semantics. Semantic completeness is not checked in partial parsing. In future research, we may consider finding a sophisticated solution of assigning confidence scores to the output RMRS fragments.

Overall though, we believe that the current 50% acceptability of segmentation is reasonable performance considering the types of noise in the speech transcript input.

As a further step to show the competence of partial parsing, we briefly investigated its application in capturing disfluent regions in speech texts. The state of the art approach in identifying disfluent regions and potentially capturing meaningful text is a shallow parsing method described in (Johnson and Charniak, 2004), which searches the text string for approximately repeated constituents. We ran their system on our random sample of the Fisher data, and compared its results to the partial parse output of the nine well-segmented partial parses analyses (every utterance of which contained some speaker-induced disfluency) to see how well partial parsing could potentially fare as an approach for identifying disfluent regions of speech text.

Often the (Johnson and Charniak, 2004) method identified disfluent regions overlapped with identified fragments found in the partial parse, the removal of which would yield a fluent sentence. As we hope to learn confidence measures to determine which fragments are contentless or repetitive in the future, we identified those partial parses where whole fragments could be deleted to obtain a fluent and meaning-preserving sentence.

In three cases, simple repeated phrases caught by (Johnson and Charniak, 2004) were also caught in some form by the partial parse partitioning. In another case, the speaker interrupts one thought to say another, and both approaches identify in a single fragment the final fluent statement. Finally, of the nine well-segmented utterances, two partial parses potentially catch deeper speaker errors that cannot be caught by (Johnson and Charniak, 2004).

5 Conclusion and Future Work

In this paper, we have presented work on partial parse selection. Different selection models have been presented and evaluated from syntactic and semantic viewpoints. In the application of spontaneous speech text processing, the method shows promising competence, as well as a few problems for further study.

One thing we did not do is a systematic comparison on the efficiency of different partial parse selection models. Although it is clear that less searching is involved with the shortest path approach and *model I* comparing to *model II*, a scientific benchmarking of such difference will be helpful for the choice between efficiency and accuracy. Also, a more sophisticated estimation of $P(\Omega|w)$ can potentially help the accuracy of the selection models.

Another alternative way of evaluation would be to generate an ungrammatical corpus by randomly introducing grammar errors. The performance of the

[3] Lexical prediction was not used here to avoid obfuscating the quality of partial parsing by introducing lexical type prediction errors.

[4] The repetition error of "it" is interpreted as a topicalization.

partial parse selection models can be measured by evaluating how much of the parsing results can be recovered from original sentences.

In the study with spontaneous speech text processing, we see a need for confidence measurement for partial analyses. We also see that the conditional probability $P(t_i|w_i)$ does not serve as a good measurement, for it largely depends on the structures that can be licensed to w_i by the grammar. This should be explored in future studies, as well.

References

Timothy Baldwin, Emily M. Bender, Dan Flickinger, Ara Kim, and Stephan Oepen. 2004. Road-testing the English Resource Grammar over the British National Corpus. In *Proceedings of the Fourth International Conference on Language Resources and Evaluation (LREC 2004)*, Lisbon.

Ted Briscoe and John Carroll. 2006. Evaluating the accuracy of an unlexicalized statistical parser on the PARC DepBank. In *Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions*, pages 41–48, Sydney, Australia.

Ted Briscoe, John Carroll, and Rebecca Watson. 2006. The second release of the RASP system. In *Proceedings of the COLING/ACL 2006 Interactive Presentation Sessions*, pages 77–80, Sydney, Australia.

Ulrich Callmeier. 2001. Efficient parsing with large-scale unification grammars. Master's thesis, Universität des Saarlandes, Saarbrücken, Germany.

John Carroll and Stephan Oepen. 2005. High efficiency realization for a wide-coverage unification grammar. In *Proceedings of the Second International Joint Conference on Natural Language Processing (IJCNLP05)*, pages 165–176, Jeju Island, Korea.

Christopher Cieri, Stephanie Strassel, Mohamed Maamouri, Shudong Huang, James Fiumara, David Graff, Kevin Walker, and Mark L iberman. 2004. Linguistic resource creation and distribution for EARS. In *Proceedings of the Rich Transcription Fall Workshop (RT-04F)*.

Ann Copestake, Dan Flickinger, Carl Pollard, and Ivan A. Sag. 2006. Minimal Recursion Semantics: an Introduction. *Research on Language and Computation*, 3(4):281–332.

Ann Copestake. 2002. *Implementing Typed Feature Structure Grammars*. CSLI, Stanford, CA.

Ann Copestake. 2006. Robust Minimal Recursion Semantics. Working Paper, Unpublished Draft 2004/2006, http://www.cl.cam.ac.uk/ aac10/papers.html.

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

Rebecca Dridan and Francis Bond. 2006. Sentence comparison using Robust Minimal Recursion Semantics and an ontology. In *Proceedings of the ACL Workshop on Linguistic Distances*, pages 35–42, Sydney, Australia.

Dan Flickinger. 2000. On building a more efficient grammar by exploiting types. *Natural Language Engineering*, 6(1):15–28.

Mark Johnson and Eugene Charniak. 2004. A tag-based noisy-channel model of speech repairs. In *Proceedings of the 42nd Meeting of the Association for Computational Linguistics (ACL'04), Main Volume*, pages 33–39, Barcelona, Spain.

Mark Johnson, Stuart Geman, Stephen Canon, Zhiyi Chi, and Stefan Riezler. 1999. Estimators for stochastic unifcation-based grammars. In *Proceedings of the 37th Annual Meeting of the ACL*, pages 535–541, Maryland.

Walter Kasper, Bernd Kiefer, Hans-Ulrich Krieger, C.J. Rupp, and Karsten Worm. 1999. Charting the depths of robust speech processing. In *Proceedings of the 37th Meeting of the Association for Computational Linguistics (ACL'99), Main Volume*, pages 405–412, Maryland, USA, June.

Robert Malouf and Gertjan van Noord. 2004. Wide coverage parsing with stochastic attribute value grammars. In *IJCNLP-04 Workshop: Beyond shallow analyses - Formalisms and statistical modeling for deep analyses*.

Robert Malouf. 2002. A comparison of algorithms for maximum entropy parameter estimation. In *Proceedings of the Sixth Conferencde on Natural Language Learning (CoNLL-2002)*, pages 49–55.

Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. 1993. Building a large annotated corpus of English. The Penn Treebank. *Computational Linguistics*, 19:313–330.

Stephan Oepen and John Carroll. 2000. Ambiguity packing in constraint-based parsing — practical results. In *Proceedings of the 1st Conference of the North American Chapter of the ACL*, pages 162–169, Seattle, WA.

Stephan Oepen, Kristina Toutanova, Stuart Shieber, Christopher Manning, Dan Flickinger, and Thorsten Brants. 2002. The LinGO Redwoods treebank: Motivation and preliminary applications. In *Proceedings of COLING 2002: The 17th International Conference on Computational Linguistics: Project Notes*, Taipei.

Miles Osborne. 2000. Estimation of Stochastic Attribute-Value Grammars using an Informative Sample. In *The 18th International Conference on Computational Linguistics (COLING 2000)*, volume 1, pages 586–592, Saarbrücken.

Stefan Riezler, Tracy H. King, Ronald M. Kaplan, Richard Crouch, John T. III Maxwell, and Mark Johnson. 2002. Parsing the Wall Street Journal using a Lexical-Functional Grammar and Discriminative Estimation Techniques. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 271–278, Philadelphia.

Kristina Toutanova, Christoper D. Manning, Stuart M. Shieber, Dan Flickinger, and Stephan Oepen. 2002. Parse ranking for a rich HPSG grammar. In *Proceedings of the First Workshop on Treebanks and Linguistic Theories (TLT2002)*, pages 253–263, Sozopol, Bulgaria.

Gertjan van Noord, Gosse Bouma, Rob Koeling, and Mark-Jan Nederhof. 1999. Robust grammatical analysis for spoken dialogue systems. *Natural language engineering*, 5(1):45–93.