WHAT'S IN A WORD GRAPH
EVALUATION AND ENHANCEMENT OF WORD LATTICES

Jan W. Amtrup *Henrik Heine* *Uwe Jost*

University of Hamburg, Computer Science Department,
Vogt-Kölln-Str. 30, D-22527 Hamburg, Germany
email: {amtrup|heine|jost}@informatik.uni-hamburg.de

ABSTRACT

During the last few years, word graphs have been gaining increasing interest within the speech community as the primary interface between speech recognizers and language processing modules. Both development and evaluation of graph-producing speech decoders require generally accepted measures of word graph quality. While the notion of recognition accuracy can easily be extended to word graphs, a meaningful measure of word graph size has not yet surfaced.

We argue, that the number of derivation steps a theoretical parser would need to process all unique sub-paths in a graph could provide a measure that is both application oriented enough to be meaningful and general enough to allow a useful comparison of word recognizers across different applications.

1. INTRODUCTION

The success of language processing modules within a speech understanding (or interpreting) system depends heavily on the quality of the outcome of the first stage in processing, the speech recognizer. A commonly used interface between a word recognizer and subsequent language understanding modules consists of a chain of words representing the sequence of words best matching the acoustic input signal (*best chain* recognition). If the word recognizer is to be used as an add-on to already existing language processing systems, this interface is the most obvious choice since the recognition results have the same form as written input (except for punctuation and capitalization).

When the recognition rate of a speech recognizer is sufficiently high, the information provided by the *best chain* may suffice for the language processing modules. In many cases, however, these modules either require perfect input or the average number of errors in the *best chain* is simply to high. In these cases, the recognizer has to pass more information to subsequent processing steps. The simplest way to do this is to present a list of different word sequences, which represent the *n-best chains* the recognizer was able to detect. Language processing modules may thus choose among the set of possible utterances presented by the recognizer.

However, it is usually not enough to deliver just the best 10 or 20 utterances, at least not for reasonable sized applications given todays speech recognition technology. To significantly increase overall system performance, n (the number of utterance hypotheses) has to be quite large. It increases exponentially with the length of the sentence [6].

Word graphs offer a simple and efficient way to represent an extremely high number of competing hypotheses and have therefore become very popular as the primary interface between speech recognizers and language processing modules [6], [2].

In order to improve speech recognition systems, a reliable measure of system performance is needed. Most well known evaluations (e.g. Resource Management, Wall-Street Journal, Switchboard) only consider the *best chain*. It is often assumed that the best system with respect to the best chain will also be the best system to produce word graphs. It is not clear, however, why this assumption should always hold. The evaluation results of the 1996 Verbmobil acoustic evaluation [8], for instance, give reason to question this assertion. Figure 1 presents the main results of the evaluation by plotting word accuracy as a function of word graph density (measured as number of hypotheses per reference word) for all participating sites. The leftmost points in all plots represent the *best chain* results. Consider the plots labeled “FP1” and “DB”, respectively. While “DB” has a significant advantage in the *best chain* class, there seems to be no difference when it comes to big word graphs.

Previous attempts to directly evaluate word graphs have been hampered by the lack of a meaningful measure of word graph quality [4]. While the notion of word accuracy can easily be extended to the best-fitting path through a word graph, it is much harder to find a meaningful measure of word graph size. In the following sections, various measures of the size and quality of word graphs are discussed and an application–oriented measure of word graph complexity is proposed.

This research was partly funded by the Federal Ministry of Education, Science, Research and Technology (BMBF) in the framework of the VERBMOBIL Project under Grant 01 IV 101 A/O.

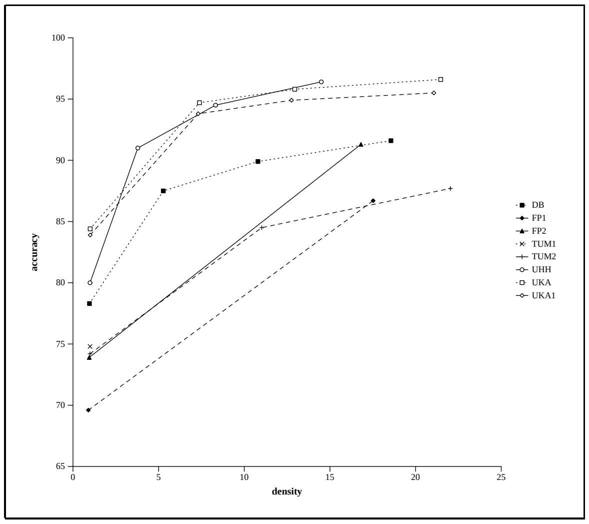

Figure 1: Results of the 1996 Verbmobil acoustic evaluation (cat. t1)

2. CONVENTIONAL GRAPH EVALUATION METHODS

A word graph is a directed, acyclic, weighted, labeled graph with distinct root and end vertices. It is a quadruple $G = (\mathcal{V}, \mathcal{E}, \mathcal{W}, \mathcal{L})$, where $\mathcal{V} = \{v_1, \ldots, v_n\}$ denotes the set of vertices, $\mathcal{E} = \{e_1, \ldots, e_m\} \subseteq \mathcal{V} \times \mathcal{V} \times \mathcal{W} \times \mathcal{L}$ the set of edges representing word hypotheses, $\mathcal{W} = \{w_1, \ldots, w_p\}$ a set of edge weights and $\mathcal{L} = \{l_1, \ldots, l_o\}$ a set of labels (usually words).

If graphs are evaluated at all (e.g. [7], [8], [6], [9]), the quality measure used is commonly a straightforward extension of the word accuracy measure as used for the best hypothesis:

$$\text{word accuracy} = 100 - 100 \cdot \frac{\text{\#errors}}{\text{\#words in transcription}}$$

$$\text{\#errors} = \text{\#substitutions} + \text{\#deletions} + \text{\#insertions}$$

The word accuracy of the graph is then just the word accuracy of the path through the graph with the best rating according to this measure.

The size (or density) of the graph is usually defined as the average number of edges per (transcribed) word (e.g. [2], [10]). This measure is rather vague, as figure 2 demonstrates. Even without any formal definition of graph size or complexity, it seems intuitively clear that the lower graph is somehow “bigger”. However, both graphs have the same number of edges.

To account for this, the average number of incoming or outgoing edges per vertex is sometimes taken into account [8]. Alternative measures proposed in [4] include the number of paths through a graph, a combination of the number of edges and the average number of outgoing edges or error rates for randomly selected paths.

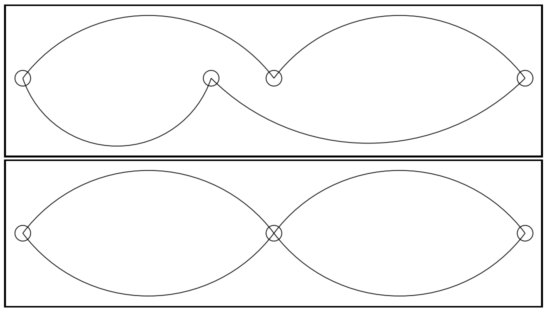

Figure 2: Example graphs

3. APPLICATION-ORIENTED EVALUATION

According to [3], “To *evaluate* is to determine what something is worth to somebody.” In order to define a meaningful measure of word graph quality or complexity, one has to determine first, who is “somebody”, i.e. what kind of language processing module is the typical customer of graph producing word recognizers. In this paper, we assume that a parser is the primary module using word graphs for further analysis. In particular, we assume that it uses a formalism based on complex features. This prevents the use of algorithms with a cubic time complexity, as we may assign different structures to each two edges or constituents covered by a rule of the syntax (e.g. by defining a feature that holds the sequence of word labels attached to the words the edge spans). We constrain ourselves to grammars that use a context-free backbone for convenience, and further constrain the rules to have at most two right-hand side nonterminals, i.e. the grammars are in Chomsky normal form. We do not, however, apply constraints regarding search strategy or pruning mechanisms.

These assumptions have serious consequences for parsing complexity, as we cannot hold a fixed size set of nonterminal categories for each interval of graph vertices (which leads to polynomial complexity). Instead, we have to produce one derivation step for each pair of edges (word hypotheses or complex edges covering more than one word) to simulate the processing of a parser.[1]

A first attempt to establish a measure for word graph size relevant for the overall performance of a speech understanding system is to investigate into the number of paths in a graph. This measure is motivated by the fact that a parser may construct an analysis for each sequence of word hypotheses covering the whole utterance. A word graph may have as many as $(\frac{|\mathcal{E}|}{|\mathcal{V}|})^{|\mathcal{V}|-1}$ paths in it, if the edges are distributed evenly among the graph. Determining the number of paths in a graph can be done efficiently in $O(|\mathcal{E}| + |\mathcal{V}|)$ time given the acyclicity of word graphs.

[1]Since a grammar may contain ambiguities, there may be more than one derivation step for each pair of edges. We abstract from this fact as well as from the fact that not every pair of edges may be combined due to grammar restrictions. Thus, we do only take into account the consequences obtainable from the raw input, the word graph, and set aside properties of grammars and such.

The next possible extension is to reduce the graph to only contain unique word sequences. The motivation behind this modification of a graph is the observation that two identically labeled yet different paths through the graph can only differ regarding two pieces of information:

- The vertices the paths visit. This should not bother a parser, since the mapping from words to exact intervals in time may be irrelevant.[2]
- The acoustic scores the words are annotated with. In this case, only the path with the highest (best) score needs to be retained.

```
[1]   for each vertex $v \in \mathcal{V}(G)$ in topological order, do
[2]       for each pair of identically labeled
              edges $e_1$, $e_2$ do
              Perform merging and create new vertex
[3]           Create a new vertex $v$ having
              $t(v) := \min(t(\beta(e_1), t(\beta(e_2))))$,
              inserting $v$ into the topological order
              Copy all edges incident from $\beta(e_1)$ to $v$
[4]           for each edge $e = (\beta(e_1), w, s, y)$ do
[5]               Create a new edge $e' := (v, w, s, y)$
              Copy all edges incident from $\beta(e_2)$ to $v$
[6]           for each edge $e = (\beta(e_2), w, s, y)$ do
[7]               Create a new edge $e' := (v, w, s, y)$
              Delete $e_1$, $e_2$
```

Figure 3: Reducing a graph to unique label sequences

Figure 3 shows the algorithm to reduce a word graph to contain unique label sequences only. It guarantees that no vertex is ever left by two edges with identical labels. This local condition has the effect that from a global point of view no two distinct paths through the graph bear identical word sequences. Unfortunately, it has a time complexity which is exponential in the number of vertices in the worst case. We introduced several optimizations (mostly concerning merging of vertices under certain conditions) which allowed us to apply this algorithm to preproduced graphs for evaluation purposes.

Just counting (unique) paths, however, ignores the form of a graph; the proportion of shared sub-paths (that need to be analyzed only once) is not taken into consideration. We therefore propose to define the complexity measure of a word graph as the number of derivations a (theoretical) parser would have to carry out in order to fully parse the graph. We first consider the analysis of one path and subsequently extend our argument to derivations over a full graph.

The number of derivation steps ($d^{(p)}$) for a full analysis of one path (p) through the graph is

$$d^{(p)} = \frac{n^3 - n}{6} \qquad (1)$$

[2]Note that this is not the case if information other than the words given by a speech recognizer is to be taken into account, e.g. prosodic information which may well be bound to specific time intervals.

where n denotes the length of the path, i.e. the number of edges covered by it. The number of derivation steps directly corresponds to the number of processing steps needed to fill the derivation matrix of the CKY-algorithm (cf. [5, p. 107]). Note again that (1) does not entail that parsing with complex feature based grammars is cubic. The only property extending over context-free parsing we use in our argument (namely not to guarantee a fix-sized set of hypotheses at any vertex) prevents us from incorporating many paths into one operation.

If we assume that all paths through the graph are independent of each other, the total number of derivations is

$$d^{(G)} = \sum_{p \in G} d^{(p)} \qquad (2)$$

which gives a linear dependency between the number of paths and the number of derivations in a graph. However, subparts of a graph are shared among different paths. Thus, the formula above is only an upper bound. To account for subgraph sharing, we have to use a slightly more complex algorithm, given in figure 4 below.

```
[1]   totalderiv $\longleftarrow 0$
[2]   for each vertex $v \in \mathcal{V}(G)$ in topological order, do
          Adjust the number of rule applications
          and the total number of derivations so far.
[3]       deriv$_v[1] \longleftarrow \#_{in}(v)$
[4]       for all $i \in \{2, \ldots, |\mathcal{V}|\}$ do
[5]           for each edge $e = (w, v, x, y)$ do
[6]               deriv$_v[i] \longleftarrow$ deriv$_v[i]$ + deriv$_w[i-1]$
[7]           totalderiv $\longleftarrow$ totalderiv + deriv$_v[i]$
[8]   return totalderiv
```

Figure 4: Determining the number of derivations in a word graph

The complexity of this algorithm yields $O(|\mathcal{E}||\mathcal{V}|)$. The method used to compute the number of derivation assumes some kind of chart parser which does not compute partial analyses twice. Shared left contexts are preserved, thus only adding once to the overall sum.

By using the number of derivations generated by this algorithm we take into account the impact of different graph shapes onto parsing effort. Thus, given two graphs with identical number of paths, the graph that has the largest amount of subgraph sharing in it will be the best one to parse. An example of an evaluation based on the number of derivation steps is shown in figure 5. It is instructive to compare it with figure 1. In figure 5, the distance between the two plots for "DB" and "UHH" seems much bigger than in figure 1 and it seems to increase with the size of the graphs. In figure 5 the word graphs delivered by "DB" appear much smaller compared to "HH" and if one imagines an interpolated plot connecting the points, the difference between the two plots appears much more constant.

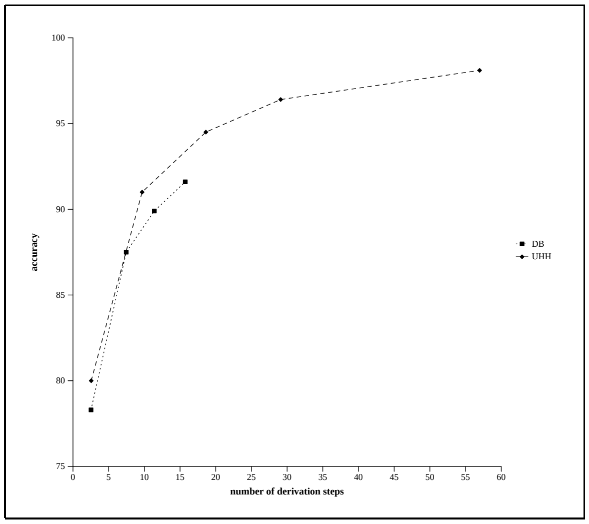

Figure 5: Recognition accuracy vs. number of derivation steps (in 10^x) (cat. t1)

4. CONCLUSION

In this paper, we have proposed a new measure for the problem size represented by word graphs. Starting from conventional evaluation of best-chain recognizers, we argued that a straightforward generalization of well known procedures to graph recognizers may be misleading. While word accuracy can be extended to graph evaluation, a sensible notion of the size of a graph is much harder to find. Measures taken until now, like the number of word hypotheses per reference word or the average fan out of vertices, are insufficient in our view, since the topology and shape of the word graphs are not properly reflected by these measures.

Instead, we propose to choose the amount of processing a hypothetical parser would have to carry out in order to process a graph as the principal measure for graph size. This measure takes into account the number of edges as well as the number of paths, and simultaneously the shape of a word graph is considered. We motivated this measure and gave efficient algorithms to compute it.

Further details considering the algorithms presented can be found in [1].

5. REFERENCES

[1] Jan W. Amtrup, Henrik Heine, and Uwe Jost. What's in a Word Graph — Evaluation and Enhancement of Word Lattices. Verbmobil Report 186, Univ. of Hamburg, December 1996.

[2] Xavier Aubert and Hermann Ney. Large Vocabulary Continuous Speech Recognition Using Word Graphs. In *ICASSP 95*, 1995.

[3] EAGLES. Evaluation of Natural Language Processing Systems. Technical Report EAG-EWG-PR.2, EAGLES Secretariat, Pisa, 1995.

[4] Michael Lehning. Evaluierung von signalnahen Spracherkennungssystemen fuer deutsche Spontansprache. Verbmobil Report 161, TU Braunschweig, 1996.

[5] Otto Mayer. *Syntaxanalyse*. Number 27 in Reihe Informatik. Bibliographisches Institut, Mannheim, 1986.

[6] Martin Oerder and Hermann Ney. Word Graphs: An Efficient Interface Between Continuous-Speech Recognition and Language Understanding. In *ICASSP93*, 1993.

[7] Erwin Paulus and Michael Lehning. Die Evaluierung von Spracherkennungssystemen in Deutschland. Verbmobil Report 70, TU Braunschweig, 1995.

[8] Joerg Reinecke. Evaluierung der signalnahen Spracherkennung. Verbmobil Memo 113, TU Braunschweig, Nov. 1996.

[9] B.H. Tran, F. Seide, and V. Steinbiss. A Word Graph Based n-best Search in Continuous Speech Recognition. In *ICSLP*, 1996.

[10] P.C. Woodland, C.J. Leggetter, J.J. Odell, V. Valtchev, and S.J. Young. The 1994 HTK Large Vocabulary Speech Recognition System. In *ICASSP95*, 1995.