ASR2000 - Automatic Speech Recognition:
Challenges for the new Millenium
Paris, France
September 18-20, 2000

A BRIEF OVERVIEW OF DECODING TECHNIQUES FOR LARGE VOCABULARY CONTINUOUS SPEECH RECOGNITION

Xavier L. Aubert

Philips Research Laboratories
Weisshausstrasse 2, 52066 Aachen, Germany
xavier.aubert@philips.com

ABSTRACT

A number of decoding strategies for large vocabulary speech recognition are examined from the viewpoint of their search space representation. Different design solutions are compared with respect to the integration of linguistic and acoustic constraints, as implied by M-gram LMs and cross-word phonetic contexts. This study is articulated along two main axes, namely, the network expansion and the search algorithm itself. Three broad classes of decoding methods are reviewed: the use of weighted finite state transducers for static network expansion, the time-synchronous dynamic-expansion search and the asynchronous stack decoding.

1. INTRODUCTION

The focus of this paper is on the representations of the search space used in a number of decoding strategies for large vocabulary continuous speech recognition (LVCSR). It appears indeed, that the specific way to handle the search network constitutes one central feature of any decoder, and reveals some of the common elements and real differences among various decoding schemes. In particular, different solutions of structuring the search space can be compared with respect to the integration of linguistic and acoustic constraints, as implied by M-gram language models (LM) and cross-word (CW) phonetic contexts.

This study has been motivated by recent advances made in LVCS decoding, concerning both the achieved level of practical performance, as well as the emergence of a new method for building a large vocabulary decoder. Near real-time capabilities are now quite common using low-cost (500 Mhz) personal computers, even for difficult tasks like broadcast news transcription (See DARPA Hub4 evaluations). Interestingly, similar levels of performance are achieved using different decoding strategies and architectures [21]. On the other hand, a full expansion of the search network is now feasible using the Weighted Finite State Transducers (WFST) framework developed at AT&T ([15]). This is quite a significant departure from the former belief that dynamic expansion could be the only viable avenue to LVCSR with long range LMs, because of the huge potential search space. Before going further, let's make the scope of the present study clear :

- The emphasis is on LVCSR using long-span LMs like trigrams. Applications dealing with very large item-lists (for example, names or addresses related to directory assistance) are not considered here (See e.g. [12]).
- References are by no means exhaustive and were chosen to illustrate some "examplative" cases. [1]
- Little attention is given to multiple-pass decoding and to the use of word-graphs. These topics are however important, being involved in almost any real system.
- Likelihood computations will not be considered here though they often represent an important fraction of the overall decoding cost, especially with continuous mixture HMMs. A number of methods can be applied to drastically reduce the complexity of the mixture density calculations (See e.g. [28]).

Hence, we focus on the "pure" search aspects and on the influence of basic design choices upon the overall complexity and efficiency of a one-pass CW M-gram decoder. This study has been articulated along two main axes, namely,

- static versus dynamic expansion of the search space,
- time-synchronous versus asynchronous decoding,

the decoder's design being affected by possible interactions between these two "dimensions" as indicated by the figure below. The paper is organized as follows. The general de-

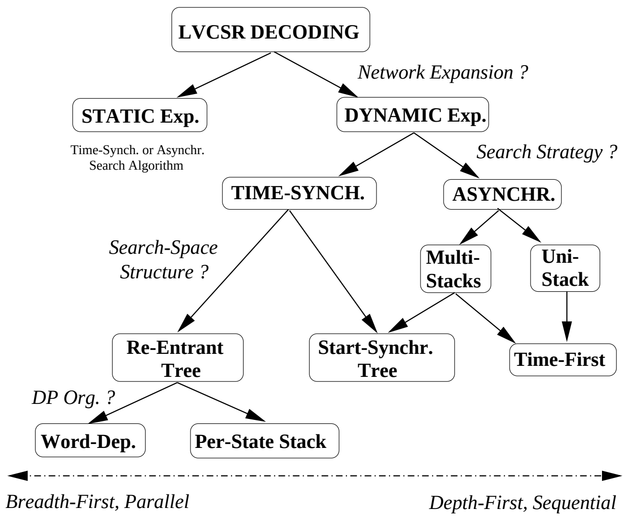

Figure 1: Classification "Tree" of Search Methods

coding problem is first specified and the full search space will be described under CW M-gram constraints. Section 4

[1] I sincerely apologize in advance to the authors who could think that their work has been overlooked.

gives a short presentation of methods that lead to a full static expansion of the search network by exploiting the inherent sparsity and redundancies of the knowledge sources. Section 5 is devoted to dynamic network expansions. Two basic approaches will be distinguished and further explained in the framework of either a time-synchronous dynamic programming (DP) search or an asynchronous stack decoder. Some comparisons will be presented regarding the (ease of) integration of cross-word contexts, long-range LMs and look-ahead pruning techniques. Finally, several general issues will be discussed in the conclusion.

2. GENERAL DECODING PROBLEM

In the Bayesian probabilistic framework, the decoding problem is specified by the well-known "simple" equation

$$\hat{W} = \underset{W}{\mathbf{argmax}} \;\; P(O|W)\, P(W) \;,$$

where $O = o_1, ..., o_t, ..., o_T$ is the time sequence of observation vectors representing the input signal, and $W = w_1, ..., w_n$ is a sequence of words within a vocabulary of size N_W. The linguistic grammar contribution is given by $P(W)$ while $P(O|W)$ contains the other lexical, phonetic and acoustic knowledge sources (KS). It is assumed that each word can be expanded in a sequence of context-dependent (CD) HMM states, possibly conditioned on the neighboring words in case of CW modeling. In practice, the Viterbi criterion is applied and, under this "maximum approximation", the search space can be described as a huge network to be explored for finding the "best" path. The recognized words of $\hat{W}$ are then determined by the most probable *state* sequence.

What makes this search problem look like a formidable task is the combinatorial nature of possible state sequences. Even after integrating all constraints introduced by the KS, the potential search space remains considerable, implying the use of complex algorithms with powerful heuristics. Consequently, in any LVCSR task, finding the *optimal* state sequence cannot be guaranteed [2] and the challenge is to get the highest accuracy from the knowledge sources, at the lowest costs in terms of memory space and computations. Broadly speaking, the task carried out by a LVCSR decoder can be decomposed in terms of five basic "actions" :

1. Generating hypothetical word sequences
2. Scoring the "active" hypotheses using the KS
3. Recombining i.e. merging paths according to the KS
4. Pruning to discard the most unpromising paths
5. Creating "back-pointers" to retrieve the best sentence

In this study, the focus will be primarily on the first and third "actions" that are the most affected by the type of search space representation, and secondarily on the points 2 and 4 while 5 will be left aside. For example, if the network has been statically expanded as a graph, generating hypotheses simply means moving to the next arcs and recombination proceeds at pre-defined nodes where the "best" scored path is selected for further extensions.

[2]The so-called search errors represent normally a very small percentage of the errors caused by using imperfect models and knowledge sources.

3. FULL CW M-GRAM SEARCH NETWORK

The structure of the search network results from the contextual constraints that are introduced by the KS at distinct levels (state, phone, word) and applied both within and across words. The scope (i.e. range of influence) of these constraints leads to the concept of early *recombination* that can be loosely formulated as : select the "best" among several paths as soon as it appears that these paths have identically scored extensions, implying that the current best path will keep dominating the others.
With a general language model $P(W)$, the search network would be an exponentially growing tree of distinct word sequences without possible recombination. However, the currently used KS have a quite limited scope, being restricted either to a few consecutive words for M-gram LMs, or to the neighboring phones for triphone HMMs. This means, for example, that only the last M-1 words of an hypothesized phrase can have an influence on the M-gram LM probability of the next word, the former preceding words being no longer taken into account. Similarly, the left conditioning context of a CW triphone at word start does not exert his influence ahead of the first phone. The "CW-phone M-gram" search space can thus be described as a finite *re-entrant* network with recombination nodes that depend on the M-1 predecessor words and the CW contexts [5]. Figure 2 shows a bigram network for a 3-word lexicon (plus a sentence-start symbol) and non cross-word CD phones[3].

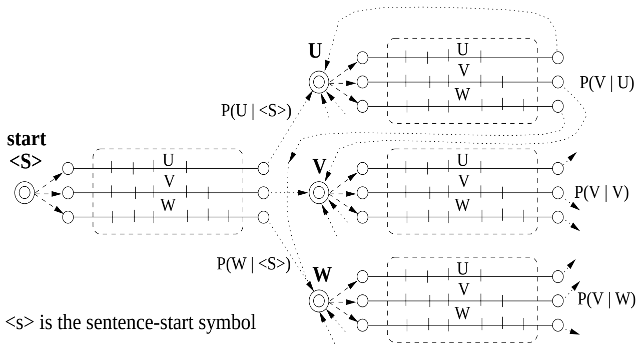

Figure 2: Word Network for Bigram LM and Non CW Models

There are $N_W + 1$ copies of the lexicon conditioned on the predecessor word and $\approx N_W{}^2$ possible word transitions. Figure 3 gives an example of inter word transitions when using either CW or non cross-word triphone contexts. For CW, the last triphone arc of the predecessor word must be replicated since it depends on the first phoneme of the following words, which is known as the CW fan-out expansion. The whole CW M-gram network is easily obtained by combining and generalizing the schemes illustrated in figures 2 and 3. The wild card symbol $*$ stands for any context.

[3]Bigram probabilities have been indicated at word-ends just for clarity.

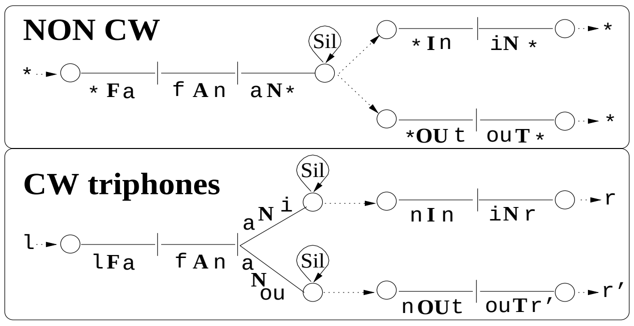

Figure 3: Cross-Word versus Non CW triphone transitions

Concerning the lexical constraints, the use of a phonetic prefix tree is widely spread since it provides a compact representation with a strongly reduced number of arcs, especially at the word beginnings where most of the search effort occurs. The prefix tree can be build from context-independent phoneme transcriptions or expanded with CD phones, the number of arcs in the first generation being then increased from a few ten to several hundred. A problem inherent to the use of a prefix tree is that word identities are only defined at leaves which delays the application of the LM probabilities. The solution consists in distributing the LM scores across the phonetic arcs, a technique called LM smearing or forwarding [37, 2, 25]. More general structures than a straightforward prefix tree can lead to larger reductions of the lexical redundancies, however, at the price of an increased complexity of both the network construction as well as the decoding algorithms (See [9, 12]).

For the handling of an M-gram search space, it is convenient to introduce a system of four dimensional coordinates representing respectively:

1. the time index
2. the language model state [4]
3. the phonetic arc
4. the acoustic state

The last two coordinates specify the position in the actual word model w.r.t. the prefix tree while the second axis determines the predecessor word history considered by the LM.

4. STATIC NETWORK EXPANSION

Expanding the whole search network prior to decoding is by no means new or recent. Actually, this has been for long the most natural approach to LVCSR until the increase of the vocabulary size in conjunction with evermore complex KS made it impractical or even impossible, due to memory limitations. The issue then became either to proceed to a "on the fly" expansion (See section 5) or to consider optimisation techniques for compacting the network.

Sparsity of Knowledge Sources and Redundancies

Two main sources of potential network compactness are :

- Exploiting the sparsity of the knowledge sources
- Detecting and taking advantage of the redundancies

[4]defined as the M-1 word history, possibly augmented with the fan-out right context for cross-words.

The sparsity of the KS stems essentially from their *regular* structure inherent to simplified model assumptions and partly from limited training data. This is especially striking for an M-gram LM but appears also true for CW position-dependent contexts generalized with CART. Let's consider a practical example of a 64k word trigram, typical of a state-of-the-art LVCSR system. Among the 4 billion of possible word bigrams, only 5 to 15 million will be included in the model and, for each of these "seen" word-pair histories, the *average* number of trigrams will be comprised between 2 and 5. Such a LM would have about 5 to 15 million of states and 15 to 90 million of arcs, requiring between 100 and 600 MB of storage. This means a reduction by seven orders of magnitude with respect to a plain 64k trigram. Concerning CW triphones, the number of distinct generalized models is typically one order of magnitude smaller than the full inventory of position-dependent contexts [5].

Coming back to M-grams, the probability of a word in an unseen context is generally obtained with an interpolation scheme involving bigram and unigram counts, for a 3G. Along this line, back-off "null" nodes have been used in several systems to take advantage of the small fraction of observed M-grams. In [4], a static tree-based representation of a bigram network has been carried out, in which the whole lexicon appears only at the "null" node. The other predecessor nodes are connected each to a much smaller subtree of the words corresponding to the bigrams seen in this context.

A clear advantage of static tree-based networks is the ease of factorizing the true M-gram probabilities which can be smeared across phone sequences, from leave to root. This results in many linear arc sequences appearing at word endings with a LM probability of one, consecutive to the forwarding process. These redundant paths can be collapsed and treated in common for all linguistic contexts of that word [4]. Other less trivial redundancies can be further reduced by applying general optimization techniques developed in the framework of finite state automaton [1].

Weighted Finite State Transducer Method (WFST)

This approach is the outcome of several research years made at AT&T and has recently reached the point of becoming an attractive alternative for building a LVCS decoder (See a.o. [32, 15, 16, 8, 7] and elsewhere in these proceedings). It offers an elegant unified framework for combining KS and producing the full search network optimized up to the HMM states. The WFST approach is briefly sketched as follows:

(1) Transducers are finite state networks associating input and output symbols on each arc possibly weighted with a log probability value. They can be used for representing various KS like lexicon with pronunciation variants, stochastic LMs or deterministic phone-in-context expansion rules.

(2) Transducers can be combined using the *composition* operator, leading to the integration of the underlying KS in one input-ouput relation. For example, C o L o G maps CD phone HMMs to words, G, L and C being transducers modeling resp. the LM, CI lexicon and CD phones.

[5]From $45^3 \times 3$ contexts to ≈ 25000 CD phone HMMs.

(3) The network is further optimized by proceeding to weighted determinization and minimization, the latter algorithm "pushing" the scores towards the initial state much alike the usual LM smearing technique [16].
Some main achievements of WFST can be summarized as:

- The KS are handled in a highly flexible way, independently of the decoder specifics about contextual range.
- The final optimized network is typically 3 to 5 times larger than the original LM.
- CW context expansion increases the network by just a few percents w.r. to the optimized CI network [15].

This last point is quite remarkable and results from postponing the context expansion after having taken advantage of the LM sparsities and lexical redundancies such that, presumably, relatively few fan-out expansions are still necessary (See figure 3).

5. DYNAMIC NETWORK EXPANSION

Integrating the network expansion in the decoding process has received considerable attention, partly from necessity because of the potential search space size, but also motivated by the self-focusing property of beam search. Applying the best KS from the outset to get the highest constraints on the search space has been the key idea leading to one-pass decoders based on dynamically build networks [14, 6, 17, 18, 23].

Another important aspect has consisted so far in assuming the *regularity* of the network structure to deal with "CW-phone M-gram" constraints. Along this line, a phonetic prefix-tree organization of the lexicon has imposed itself as a generic building block of the network, since it offers a good tradeoff between simplicity and compactness at word start.
Two basic approaches for dynamically exploring an M-gram tree-structured network can be distinguished:

- The *re-entrant tree* [6] where the word history remains attached to each path coming back to the root.
- The *start-synchronous tree* [7] which generates the next hypotheses of all paths having simultaneous word ends.

Both methods have been applied in the framework of time-synchronous dynamic programming (DP) search while the second has been used in multi-stack asynchronous decoders.

Time-Synchronous Search based on a Re-Entrant Tree

This appears like the most popular technique used for LVCSR probably because of the conceptual simplicity of DP that leads to a straightforward implementation of beam pruning, and because the network expansion strategy closely follows the static counterpart described in section 3. One important feature is that word-boundary optimization is carried out by the DP recurrence when re-entering the tree [27]. For a correct use of M-gram LMs with $M >= 2$, it is necessary to keep track of the individual M-1 word histories of the active paths, until the recombination step can take place at the next word ending.[8] Two search organizations have been pursued with a re-entrant tree to fulfill this "optimality" criterion and these can be explained with the coordinate system introduced at the end of section 3. Differences mainly concern the *order* with which the last three search-space coordinates are spanned, time being here the independent variable.

The Word-History Conditioned DP Organization [18]
The partially expanded hypotheses are recorded in lists and grouped according to the (same) word-history. This means that the three coordinates are spanned in the following order: LM-State $\rightarrow$ Arc-Id $\rightarrow$ State-Id, the emphasis being on the common predecessor word context.

The Per-State Stack Organization [2, 3]
This method focuses on the multiple instantiations of the same HMM state that happen when a phonetic arc is simultaneously active in different LM contexts. The ordering of the coordinates becomes : Arc-Id $\rightarrow$ State-Id $\rightarrow$ LM-State where the word-history appears now in the last position.
Compared to the word-history conditioning, the per-state stack offers some more flexibility for pruning at state level, possibly with less "strict"LM recombination rules [3, 10].

Integration of Cross-Word Contexts
In time-synchronous dynamic search, CW modeling implies that multiple contexts have to be considered at word end to *anticipate* for the next successor words. This leads to the fan-out expansion [23] where the last phonetic arc is given several instances depending on the *right* conditioning context. This step, also done dynamically, can be sped-up by (1) taking advantage of the redundancy among CD phone HMMs [36] and (2) applying a special LM look-ahead pruning scheme [5, 29]. Next, CW expansion implies a modification of the word-end recombination nodes since the identity of the fan-out right context restricts the set of word successors [5]. This in turn requires selecting the compatible arcs at tree root. Last, when re-entering the tree, the left context of the phones in the first generation is specified by the last phone of the previous word, implying either an on-the-fly CD HMM instantiation or multiple instances of the first tree generation. Figure 4 illustrates the main lines of these CW transitions in the framework of a re-entrant prefix tree.

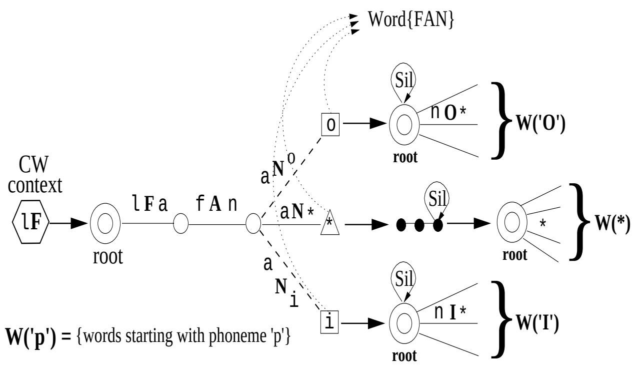

Figure 4: Cross-Word Transitions with an Optional "Long" Pause

[6]Also known as "word-conditioned copies of the lexical tree" [18].

[7]As suggested by [33], but also named as "The time-conditioned approach in dynamic programming search" [30].

[8]Sub-optimal recombinations can also be used when suitable (e.g. [20]).

Time-Synchronous Search & Start-Synchronous Trees

The main idea is to share a single one-word extension step among paths ending at the same "current" time, however, with different LM histories. In this way, the DP time-alignment process is done at most once per word model and per start time, avoiding the need for so-called LM word copies. The search space is thus structured on the start time of the successor words which are hypothesized by propagating the DP recurrence across one prefix tree started at that time. As a consequence, the word-boundary optimization is no longer performed implicitly as for the re-entrant tree, and has to be carried out in a separate step occurring at each word-ending. This is beneficial for generating dense word graphs [24, 19] but makes the LM recombination step (much) more expensive [26]. This is however compensated by the size of the activated search space which appears almost independent of the LM complexity. A remarkable feature is that the average number of active tree hypotheses per time frame is about the average word duration [26, 30].

On the other hand, this architecture appears less favorable for CW context expansions and for LM look-ahead pruning schemes involving more than just unigram probabilities[9]. This is a direct consequence of sharing the prefix-tree expansion among simultaneously ending words bearing different LM and phonetic (left) contexts, as opposed to the re-entrant tree where paths with distinct histories are treated separately.

Asynchronous Stack Decoding

This approach stems from sequential decoding as developed in communication theory [14]. In LVCSR, stack decoding implements a best-first tree search which proceeds by extending word by word, one or several selected hypotheses *without* the constraint that they all end at the same time. Running hypotheses are handled using a *stack* which is a priority queue sorted on likelihood scores and possibly on time. Depending on the implementation, there may be one single stack [31] or several stacks [13], each one grouping the theories associated to the same end-time[10].

Compared to time-synchronous beam search, there are three specific problems to be solved in a stack decoder:

- Which theory(ies) should be selected for extension ?
- How to efficiently compute one-word continuations ?
- How to get "reference" score values for pruning ?

The first point relates to the use of heuristics (known as A^* principle), and essentially depends on which information is available regarding the not yet decoded part of the sentence. In a multi-pass strategy, a first decoding can provide an estimation of the probability for the "remaining" part. An alternative that does not need looking-ahead in the signal has been presented in [31] and rely on least upper bounds taken on the path scores that have been expanded so far. In practice, this leads to a "shortest best path" choice [11, 33, 38].

[9]So far, this method has been tested only with unigram smearing and non crossword models, to the best of my knowledge.

[10]Distinguishing between uni- and multi-stack techniques will not be further considered here, being outside the scope of this text.

The one-word extensions can either be computed with the start-synchronous tree method of previous section [13] or using a fast-match [11] to first get a short list of word candidates that are processed sequentially for continuing one (or more) theory [11, 22]. Interestingly, the start-synchronous tree exploration is done either by standard time-synchronous DP [13] or by a "time-first" asynchronous method [34].

Pruning is non trivial due to the difficulty of comparing scores of paths having different lengths. The solution consists in progressively updating the best likelihood scores that can be achieved along the time axis by a path having complete word extensions. This requires storing temporarily the score sequences of the respective paths. Broadly speaking, this leads to the concept of *envelope* defined as the lowest upper bound of the individual score "profiles" of the paths expanded so far [11]. Based on the current score envelope, a path may be labeled as active or not and this decision may be reconsidered in the course of the decoding process.

Last, integrating CW phonetic contexts is easily achieved in two stages [11, 35] by considering left-only conditioned contexts first, and after the one-word extension, proceeding to a rescoring with the now available right context. The main advantage w.r.t. time-synchronous search is that CW modeling can be carried out on individual word strings without the need of fan-out expansions and with great ease for taking longer context ranges into accounts (at least to the left). Likewise are long-range LMs easily integrated, recombination being subjected to the known dominance principle.

6. CONCLUSION

In attempting to summarize this short tour about decoding techniques, the following pros and cons could be suggested:
(1) Static network expansion using WFST appears quite elegant, leads to a very flexible decoding environment and integrates CW contexts as well as grammar spreading at almost no overhead costs. On the other hand, this avenue strongly relies on current KS sparsities which might be a handicap.
(2) Time-synchronous dynamic search achieves very efficient pruning and recombination schemes in a conceptually simple and unified framework. The handling of LM copies has proven surprisingly efficient. Integrating CW models is however delicate if not "tricky" and does not seem to "scale" easily to larger contexts. This approach might also take advantage of exploiting some of the M-gram sparsities.
(3) Stack decoders require assembling a number of complex algorithms and, due to less favorable pruning conditions, may be more influenced by fast match performance. However, they offer an ideal decoupling w.r.t. the LM interface and are well-suited for integrating longer contextual constraints of whatever nature.

ACKNOWLEDGMENTS

I would like to thank my colleagues at Philips Research in Aachen for their support, especially Christoph Neukirchen,

[11]Time-synchronous search can of course also benefit from acoustic look-ahead pruning, typically done at the phoneme level [18, 3].

Hans Dolfing and Matthew Harris, with whom I got many enlightening discussions while preparing this overview.

REFERENCES

[1] Aho A.V., Hopcroft J.E. and Ullman J.D., "The Design and Analysis of Computer Algorithms", Addison Wesley, 1974.

[2] Alleva, F., Huang, X. & Hwang, M.-Y., "Improvements on the pronunciation prefix tree search organization", Proceedings of ICASSP, pp. 133-136, Atlanta, GA, 1996.

[3] Alleva Fil, "Search Organization in the Whisper Continuous Speech Recognition System", in Proceedings of IEEE ASRU Workshop, pp 295–302, Santa Barbara, CA, Dec. 1997.

[4] Antoniol G., Brugnara F., Cettolo M. and Frederico M., "Language Model Representations for Beam-Search Decoding", in Proc. ICASSP pp. 588–591, Detroit, MI, May 1995.

[5] Aubert X., "One Pass Cross Word Decoding For Large Vocabularies Based On A Lexical Tree Search Organization", Proc. Eurospeech, pp. 1559-1562, Budapest, Hungary, 1999.

[6] Bahl L. R., Jelinek F., Mercer R. L., "A Maximum Likelihood Approach to Continuous Speech Recognition", IEEE Trans. on Pattern Analysis and Machine Intelligence, Vol. 5, pp. 179-190, March 1983.

[7] Bazzi Issam and Glass James, "Heterogeneous Lexical Units for Automatic Speech Recognition: Preliminary Investigations", Proc. ICASSP pp. 1257-1260, Istanbul, Turkey, 2000.

[8] Boulianne G., Brousseau J., Ouellet P., Dumouchel, P., French Large Vocabulary Recognition with Cross-Word Phonology Transducers", in Proc. ICASSP. pp. 1675–1678, Istanbul, Turkey, May 2000.

[9] Demuynck K., Duchateau J. and Van Compernolle D., "A static lexicon network representation for cross-word context-dependent phones", in Proc. Eurospeech, Vol. 1, pp. 143–146, Rhodos, Greece, Sept. 97.

[10] Finke M., Fritsch J., Koll D. and Waibel A., "Modeling and Efficient Decoding of Large Vocabulary Conversational Speech", in Proc. Eurospeech, Vol. 1, pp. 467-470, Budapest, Hungary, Sept. 1999.

[11] Gopalakrishnan P.S., Bahl L. R.,and Merver R.L., "A Tree Search Strategy for Large-Vocabulary Continuous Speech Recognition", Proc. ICASSP, pp. 572-575, Detroit, WA, 95.

[12] Hanazawa K., Yasuhiro M. & Furui S., "An efficient search method for large-vocabulary continuous-speech recognition", Proc. ICASSP, Vol. 3, pp. 1787-1790, Germany, 1997.

[13] Hochberg M., Renals S. Robinson A. and Kershaw D., "Large Vocabulary Continuous Speech Recognition using a Hybrid Connectionist-HMM System", in Proc. ICSLP, pp. 1499-1502, Yokohama, Japan, Sep. 1994.

[14] Jelinek F., Bahl L. R. and Mercer, R. L., "Design of a Linguistic Statistical Decoder for the Recognition of Continuous Speech", IEEE Trans. Inf. Th. Vol IT-21, pp. 250-256, 1975.

[15] Mohri, M., Riley M., Hindle, D., Ljolje A., Pereira F., "Full Expansion of Context-Dependent Networks in Large Vocabulary Speech Recognition", in Proc. ICASSP, pp. 665-668, WA, Seattle, May 1998.

[16] Mohri, M. and Riley M., "Network Optimizations for Large-Vocabulary Speech Recognition", in Speech Communication Journal, Vol. 28, Nr. 1, pp. 1–12, May 1999.

[17] Ney H., Mergel D., Noll A. and Paeseler A., "A Data Driven Organization of the Dynamic Programming Beam Search for Continuous Speech Recognition", in Proc. ICASSP, pp. 833-836, Dallas, Texas, April 1987. Also in IEEE Trans. on Signal Processing, Vol. SP-40, No. 2, pp. 272-281, Feb. 1992.

[18] Ney, H., Haeb-Umbach, R., Tran, B.-H. & Oerder M., "Improvements in beam search for 10000-word continuous speech recognition", pp. 13–16 in Proceedings of IEEE Int. Conf. on Acoustics, Speech and Signal Processing, Vol.I, pp. 9-12, San Francisco, CA, March 1992.

[19] Ney H., Ortmanns S. and Lindam I., "Extensions to the Word Graph Method for Large Vocabulary Continuous Speech Recognition", Proc. ICASSP, pp. 1787-1790, Munich, 1997.

[20] Nguyen Long and Schwartz Richard, "The BBN Single-Phonetic-Tree Fast-Match Algorithm", pp. 1827–1830, Proc. ICSLP, Sydney, Australia, Nov. 98.

[21] NIST, "Dec'99 evaluation for 10x real-time transcriptions of Broadcast News", to appear in DARPA Proceedings, 2000.

[22] Novak M. and Picheny M., "Speed Improvement of the Time-Asynchronous Acoustic Fast Match", pp. 1115–1118, Proc. Eurospeech'99, Budapest, Hungary, Sep. 1999.

[23] Odell J.J., Valtchev V., Woodland P.C. & Young S.J., "A One Pass Decoder Design for Large Vocabulary Recognition", in Proceedings of ARPA Spoken Language Technology Workshop, Plainsboro, NJ, pp. 405-410, March 1994.

[24] Oerder, M. & Ney H., "Word Graphs : An Efficient Interface Between Continuous Speech Recognition and Language Understanding"', in Proc. ICASSP, Vol. 2, pp. 119-122, Minneapolis, MN, April 1993.

[25] Ortmanns S., Ney H. & Eiden A., "Language-Model Look-Ahead for Large Vocabulary Speech Recognition", in proc. ICSLP 96, pp. 2095-2098, Philadelphia, October 1996.

[26] Ortmanns S., Ney H., Seide F. and Lindam I., "A Comparison of Time Conditioned and Word Conditioned Search Techniques for large vocabulary speech recognition", in proceedings of ICSLP 96, pp. 2091-2094, Philadelphia, Oct. 1996.

[27] Ortmanns S., Ney H. & Aubert, X., "A word graph algorithm for large vocabulary continuous speech recognition", in Computer Speech and Language, Vol. 11, pp. 43-72, 1997.

[28] Ortmanns, Stefan, Firzlaff Thorsten and Hermann Ney, "Fast Likelihood Computation Methods for Continuous Mixture Densities in Large Vocabulary Speech Recognition", pp. 139–142 in Proc. Eurospeech'97, Rhodes, Greece, Sep. 1997.

[29] Ortmanns S., Reichl W. and Chou Wu, "An Efficient Decoding Method for Real-Time Speech Recognition", in Proc. Eurospeech, pp. 499-502, Budapest, Hungary, Sep. 1999.

[30] Ortmanns Stefan and Ney Hermann, "The Time-Conditioned Approach in Dynamic Programming Search for LVCSR", to appear in IEEE Trans. of Speech and Audio Processing.

[31] Paul, Douglas B., "An efficient A^* Decoder Algorithm for Continuous Speech Recognition with a Stochastic Language Model", in Proc. ICASSP, Vol. 1, pp. 25-28, CA, March 1992.

[32] Pereira F., Riley M. and Sproat R., "Weighted Rational Transductions and their Application to Human Language Processing" in Proc. of ARPA HLT Workshop, March 1994.

[33] Renals Steve and Hochberg Mike, "Start-Synchronous Search for Large Vocabulary Continuous Speech Recognition" , IEEE Tr. Sp. Audio Proc., Vol. 7-5, pp. 542-553, 1999.

[34] Robinson Tony and Christie James, "Time-First Search for Large Vocabulary Speech Recognition", in Proc. ICASSP, pp. 829–832, USA, Seattle, April 1998.

[35] Schuster Mike, "Memory-efficient LVCSR search using a one-pass stack decoder", in Computer Speech and Language (2000), Vol. 14, pp. 47-77, 2000.

[36] Sixtus, A., Molau S., Kanthak S., Schlüter R. and Ney H., "Recent Improvements of the RWTH Large Vocabulary Speech Recognition System on Spontaneous Speech", in Proc. ICASSP, pp. 1671-1674, Istanbul, Turkey, May 2000.

[37] Steinbiss V., Tran B.-H., Ney, H., "Improvements in Beam Search", in Proceedings of ICSLP, pp. 2143-2146, Yokohama, Japan, Sep. 1994.

[38] Willett D., Neukirchen Chr., and Rigoll G., "Ducoder - The Duisburg University LVCSR Stack Decoder", in Proc. ICASSP, pp. 1555-1558, Istanbul, Turkey, May 2000.