Computer Speech and Language (2002) **16**, 89–114
doi: 10.1006/csla.2001.0185
Available online at http://www.idealibrary.com on IDEAL®

An overview of decoding techniques for large vocabulary continuous speech recognition

Xavier L. Aubert†

Philips Research Laboratories, Weisshausstrasse 2, 52066 Aachen, Germany

Abstract

A number of decoding strategies for large vocabulary continuous speech recognition (LVCSR) 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 language models (LM) and cross-word (CW) phonetic contexts. This study is structured along two main axes: the network expansion and the search algorithm itself. The network can be expanded statically or dynamically while the search can proceed either time-synchronously or asynchronously which leads to distinct architectures. Three broad classes of decoding methods are briefly reviewed: the use of weighted finite state transducers (WFST) for static network expansion, the time-synchronous dynamic-expansion search and the asynchronous stack decoding. Heuristic methods for further reducing the search space are also considered. The main approaches are compared and some prospective views are formulated regarding possible future avenues.

© 2002 Academic Press

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 of handling the underlying 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 large vocabulary 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. Interestingly, similar levels of performance are achieved using quite different decoding strategies and architectures (DARPA, 2000). In addition, a full expansion of the search network has been shown to be feasible using the weighted finite state transducers (WFST) framework developed at AT&T (Mohri, Riley, Hindle, Ljolje & Pereira, 1998). This is quite a significant departure from the former belief that dynamic expansion could be

†E-mail: xavier.aubert@philips.com

0885–2308/02/010089 + 26 $35.00/0 © 2002 Academic Press

the only viable approach to LVCSR with long range LM because of the huge potential search space, and this in turn deserves our attention to understand what made this evolution possible.

Before going further, the scope of the present study has to be made clear:

- The emphasis is on LVCSR using long-span LM like trigrams. Applications dealing with very large item lists (for example, names or addresses related to directory assistance) are not considered here (see, for example, Hanazawa, Yasuhiro & Furui, 1997).
- References are by no means exhaustive and were chosen to illustrate some "prototypical" cases.[1]
- Little attention is given to multiple-pass decoding and to the use of word-graphs. These topics, while clearly important in developing a recognizer, deserve a more specific study than can be given here.
- Likelihood computations will not be considered here though they often represent an important part of the overall decoding cost, especially with mixtures of continuous distributions. A number of methods can be applied to drastically reduce the complexity of the mixture density calculations (see, for example, Ortmanns, Firzlaff & Ney, 1997).

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 structured along two main axes, namely,

- static vs. dynamic expansion of the search space;
- time-synchronous vs. asynchronous decoding.

As will be shown in the following, the decoder's architecture is deeply shaped by the interactions between these two main lines.

The paper is organized as follows. The general decoding problem is first formulated in the Bayesian probabilistic framework for hidden Markov models (HMM) and the concept of early recombination is introduced because of its key role in the efficient search for the "best" state sequence. The main actions that have to be carried out by any LVCSR decoder are described. The usual knowledge sources involved in a large vocabulary CW m-gram decoder are reviewed in Section 3, including the use of a phonetic prefix tree. This leads to the representation of the whole search space in terms of a finite re-entrant network. A convenient coordinate system is also introduced. Section 4 presents a broad classification of decoding methods in a tree-like diagram, based on the main axes of network expansion and search strategy. The following two sections are devoted to a review of the principal decoding approaches: Section 5 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 and Section 6 is devoted to *dynamic* network expansion techniques. Two basic ways for dynamically exploring a tree-structured m-gram network can be distinguished. These are the re-entrant lexical tree method and the start synchronous tree method emphasizing, respectively, the role of the word *linguistic contexts* and of the word *start times*. These two search avenues are further explained in the framework of either a time-synchronous dynamic programming (DP) search or an asynchronous stack decoder. In Section 7, two heuristic methods are briefly described for reducing further the size of the search space beyond standard beam pruning capabilities, either by constraining the word start times[2] or by looking ahead into the acoustic content of the signal. Section 8 addresses the methodology suitable for evaluating a "real" decoder and presents some experimental evidence drawn from recent DARPA

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

[2] The so-called word-pair approximation is one example of such technique.

evaluation results for broadcast news transcription. It appears that there is no clear dominant method so far and that the success of any decoder lies in the optimized integration of several components. Finally, a number of pros and cons of the main search approaches are proposed in the conclusion and several directions are considered regarding what could be promising avenues for further improvements in large vocabulary decoding.

2. General formulation of the decoding problem

2.1. Bayesian framework and Viterbi approximation

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

$$\hat{W} = \underset{W}{\textbf{arg max}} \ \{P(O|W)P(W)\},$$

where $O = o_1^T = o_1, \ldots, o_t, \ldots, o_T$ is the time sequence of observation vectors representing the *input* signal, and $W = w_1^n = w_1, \ldots, w_n$ 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. It is assumed that each word can be expanded in a sequence of context-dependent HMM states, possibly conditioned on the neighbouring words in case of CW modelling, which leads to the equivalent equation

$$\hat{w}_1^n = \arg\max_{w_1^n} \left\{ P(w_1^n) \cdot \sum_{s_1^T} P(o_1^T, s_1^T | w_1^n) \right\}$$

where s_1^T stands for any HMM state sequence of length T being emitted by the word sequence w_1^n, each such sequence contributing, in general, to the overall sentence probability. 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 according to

$$\hat{w}_1^n = \arg\max_{w_1^n} \left\{ P(w_1^n)^\alpha \cdot \underset{s_1^T}{\textbf{Max}} \ [P(o_1^T, s_1^T | w_1^n)] \right\}$$

where the heuristic LM factor α has now been introduced as well. The recognized word sequence $\hat{w}_1^n$ is thus determined by the most probable *state* sequence in the underlying search network. What makes this decoding problem a formidable task is the combinatorial nature of possible state sequences implying the use of complex algorithms with powerful heuristics like beam pruning, an exhaustive search being intractable for almost any LVCSR task.

2.2. Recombination principle

The structure of the search network results from the contextual constraints that are introduced by the knowledge sources at distinct levels (state, phone, word) and are applied both within and across words. As we will see, the scope (i.e. the range of influence) of these constraints is quite limited in the currently used knowledge sources characterized by a "short memory". This restricted scope allows for an important optimization of the search process based on the concept of early *recombination* that can be formulated as:

Select the "best" among several paths in the network as soon as it appears that these paths have identically scored extensions, implying that the current best path will keep dominating the others.

In other words, do not waste computational resources by extending partial theories that have no chance of leading to the best sentence anymore! The action of recombining hypotheses is also often named *path merging*. As an example, with a general LM $P(W)$, the search network would be an exponentially growing tree of distinct word sequences without possible recombination. However, the currently used m-grams rely on probabilities that depend only on the last m–1 words of the hypothesized phrase, the former preceding words being no longer taken into account by the LM.

Recombination is inherent to the dynamic programming formulation (Bellman, 1957) and is also important in the A^* search algorithm used for sequential decoding (Nilsson, 1998) where its role has been recently made more explicit.[3] Consequently, identifying promptly the recombination nodes is a must towards efficiency and this might be non-trivial in some cases, for example, when a CW network is dynamically expanded.

2.3. Main actions to be performed by any decoder

Broadly speaking, the task to be carried out by a large-vocabulary continuous-speech decoder can be decomposed in terms of five basic "actions":

1. Generating hypothetical word sequences, usually by successive extensions.
2. Scoring the "active" hypotheses using the knowledge sources.
3. Recombining i.e. merging paths according to the knowledge sources.
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. For example, if the search 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. The second and fourth points are shortly addressed in Section 7 while the fifth point will be left aside.

3. Representation of the knowledge sources

The knowledge sources used in automatic speech recognition are naturally structured in a four-level hierarchy with, successively, the acoustic HMM states, the phones in context, the word models and finally, the sequences of words or phrases up to the sentence level. The specifics of HMM states and emission probability distributions will be left aside since they do not directly interact with the search network[4] and we will only be concerned with the other three modelling levels.

3.1. Use of stochastic m-gram LM

The role of the LM is to introduce constraints upon the word sequences expressing the syntactic and semantic characteristics of the linguistic domain at hand. The use of a *stochastic* m-gram LM has two clear implications:

[3]The situation where the graph to be searched could not be a tree is considered on p. 142 of Nilsson (1998).

[4]Apart from the occurrence of model copies due to parameter tying as considered in Section 5.1.

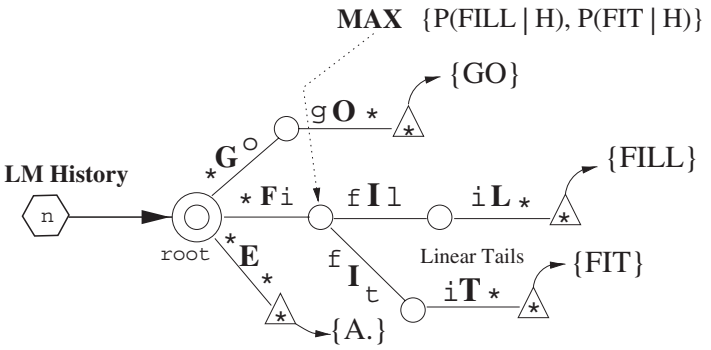

Figure 1. Prefix tree structure of the lexicon with LM look-ahead.

1. the search network is fully branched at the word level, each word being possibly followed by any other;
2. the word probabilities depend on the $m-1$ predecessors only:

$$P(w_n|w_{n-1}, w_{n-2}, \ldots, w_2, w_1) = P(w_n|w_{n-1}, \ldots, w_{n-m+1}).$$

Consequently, the recombination nodes at word ends are defined in terms of the $m-1$ predecessors and the search process must somehow keep track of the individual $m-1$ word histories, until the optimization step can take place. How this is precisely achieved and implemented depends on the actual design of the decoder and will be explained in the next sections. As already mentioned, postponing the recombination step beyond the $m-1$ word phrases would be useless and inefficient. However, merging concurrent theories *before* this point would be sub-optimal with a high risk of making search errors by losing the best path due to a wrong integration of the grammar constraints. We will come back to this point in Section 7.

3.2. Prefix-tree organization of the lexicon

The lexicon defines the list of words with their phonetic transcriptions in terms of a small number of context-independent phoneme symbols. Some word entries may have multiple pronunciations, possibly with prior (unigram) probabilities associated to their occurrence frequencies (Schramm & Aubert, 2000).

Structuring the lexicon as a phonetic *prefix tree* is widely applied since it provides a compact representation with a reduced number of arcs, especially at the word beginnings where most of the search effort occurs. Indeed, when using a stochastic m-gram LM each word of the lexicon is a possible successor to every hypothesized word end. Hence, sharing the common word stems results in a dramatic reduction of the number of phone arcs needed to generate the next word startup hypotheses. The prefix tree can be built from context-independent phoneme transcriptions or expanded with context-dependent phones (like triphones), the number of arcs in the first generation being then increased from a few tens to several hundreds.

A problem inherent with the use of a prefix tree is that word identities are only *defined* at leaves which delays the integration of the LM probabilities (see Fig. 1). The solution consists in factorizing the word probabilities across the phonetic arcs, a technique called LM smearing or forwarding (Steinbiss, Tran & Ney, 1994; Alleva, Huang & Hwang, 1996;

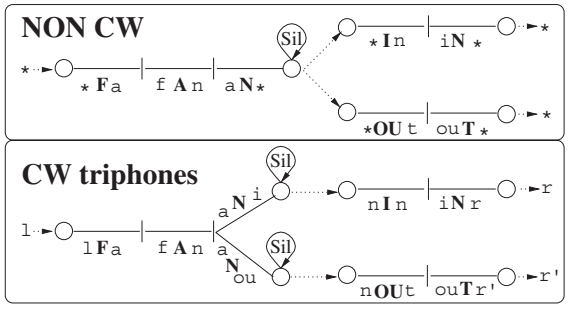

Figure 2. CW vs. non-CW triphone transitions.

Ortmanns, Ney & Eiden, 1996). This achieves a double effect of (1) anticipating the word probabilities and (2) smoothing the score distributions across the phonetic arcs, both factors being highly beneficial for the pruning efficacy. Another interesting side effect consecutive to LM smearing concerns the handling of the so-called "linear tails" in the prefix tree. These phonetic arc sequences appear when a given word no longer shares any arc with another word such that the factorized LM probabilities are equal to one. These tails can be merged across all linguistic contexts and this happens at any point of the lexical tree from which a word identity is uniquely determined.

More general structures than a straightforward prefix tree can lead to larger reductions of the lexical redundancies, for example, by merging both the identical word beginnings and endings into a phonetic *network*.[5] However, the construction of such a network is more complex and the decoding algorithm has to be modified since word identities are no longer uniquely defined at leaves as happens in a prefix tree, which also has some influence on the smearing of LM scores (see Demuynck, Duchateau & Van Compernolle, 1997; Hanazawa *et al.*, 1997).

3.3. Context-dependent phonetic constraints

The use of *context-dependent* phone models is not indifferent regarding the search space especially at the junction between successive words. Figure 2 gives an example of inter word transitions when using either CW or only within-word triphone contexts. With CW models, 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 fan-out expansion. The wild card symbol $*$ stands for any context and non-cross-word triphones degenerate to left or right diphones at a word boundary.

The constraints introduced by context-dependent phones have a quite limited scope too, being restricted to just the neighbouring phones for triphone HMMs. This means, for example, that the left conditioning context of a CW triphone at word start does not exert its influence ahead of the first phone. If only linear sequences of words are considered by the decoder then, of course, the use of CW contexts makes no difference in terms of the size of the search space. In the general case, a word end hypothesis may be followed by several or all lexical entries. Consequently, the definition of recombination nodes must be extended to take account of the right context r of the fan-out arc since the subset of words that may follow will be restricted to the words starting with phoneme r (Aubert, 1999).

[5]Sharing common suffixes is especially effective when the lexicon includes across-word fan-out arcs.

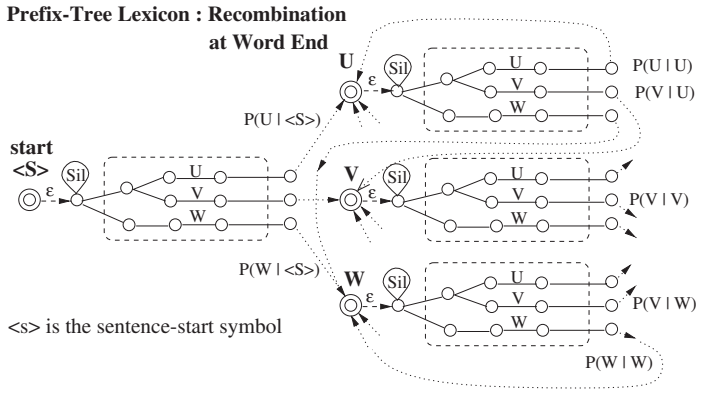

Figure 3. Prefix-tree network for a bigram and non-CW models.

3.4. Search-space coordinate system of four dimensions

Following the presentation of the knowledge sources given in the previous sections, the search space of a general "CW m-gram" decoder can be described as a finite re-entrant network with recombination nodes that depend on $m-1$ predecessor words and possibly on the fan-out right contexts (Aubert, 1999). The corresponding search space can be spanned by means of four-dimensional coordinates representing, respectively:

1. the time index,
2. the LM state,
3. the phonetic arc,
4. the acoustic state,

where the LM state is defined as the $m-1$ word history, possibly augmented with the fan-out right context for CW models. Depending on the search strategy, the time coordinate will be the independent variable or not. The last two coordinates specify the position in the actual word model with respect to the prefix tree organization while the second axis determines the predecessor word history considered by the LM. This coordinate system is quite useful for describing some organizational aspects of the search algorithms described in Section 6.

3.5. Illustration of a bigram non-CW search network

Considering a three-word lexicon $\{U, V, W\}$ plus a sentence-start symbol $\langle s \rangle$, the bigram search network can easily be expanded for non-CW phones. The impact of the lexical organization is quite interesting and deserves some comments. If the lexicon is structured as a *prefix tree*, the LM recombination has to be postponed up to the word *ends* which leads to the need for copies of the lexical tree in distinct m-gram contexts, as illustrated in Fig. 3. The bigram tree network requires $N_W + 1$ copies of the lexicon conditioned on the predecessor words and there are $\approx {N_W}^2$ possible word transitions. In the general m-gram case ($m \geq 1$), the total number of prefix tree copies is equal to ${N_W}^{m-1}$ where N_W is the lexicon size and m the LM order.

On the other hand, when using a (flat) linearly-structured lexicon, the LM recombination can be performed at the word startups using empty transitions bearing the m-gram probabilities and, in the *bigram* case, there is no need of lexical copies as shown in Fig. 4. In the general m-gram case ($m \geq 2$), the total number of copies of the linear lexicon would now be ${N_W}^{m-2}$.

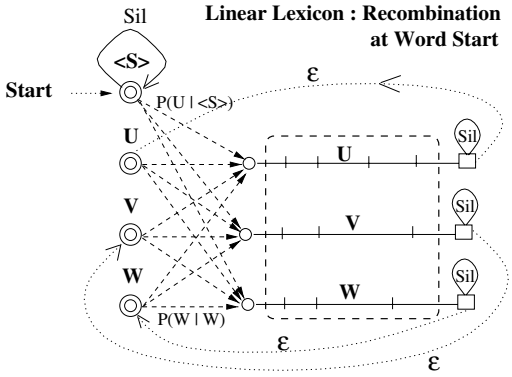

Figure 4. Bigram network using a linear lexicon and non-CW models.

Compared to the prefix tree case, the number of word arcs is smaller by a factor equal to the vocabulary size. However, the decoding cost using a beam search technique would be much larger with a linear lexicon. This is due to the fact that, the common word stems not being shared, a much larger number of arcs will be activated at each word end hypothesis. This shows that minimizing the total number of arcs in the static expansion of a search network is not necessarily the best criterion from the viewpoint of the computational decoding costs, even if it reduces the storage requirements. This apparent paradox is solved when considering the degree of determinization of the networks as defined in finite state automata theory (Aho, Hopcroft & Ullman, 1974): the linear lexicon network is definitely non-determinized (see Section 5.3). The whole "CW *m*-gram" network can be obtained by combining and generalizing the schemes illustrated in Figs 2 and 3.

4. Broad classification of decoding methods

Before proceeding to a brief description of several basic algorithms for large vocabulary continuous speech decoding, a (tentative) classification is first proposed by means of a decision-tree. This, however, requires anticipating somehow what will be explained in the next sections and, therefore, the reader is likely to come back to Fig. 5 later on.

The first two questions concern, naturally, the network expansion mode and the search strategy itself which constitute the main axes of the present study. If the decoder relies on a static expansion of the network prior to decoding, then any search algorithm may be applied though in practice a straightforward time-synchronous Viterbi search is often adopted. If the network expansion is done dynamically "on the fly", the choice of either a time-synchronous or an asynchronous search strategy appears determinant. The second choice leads to a stack decoding approach which may involve one or several stacks (i.e. sorted list of running theories). In both cases, the emphasis is placed on a *sequential* expansion of individual word sequences along the time axis which leads to the most "depth-first"-like search strategy as indicated by the low horizontal axis.

Coming back to the time-synchronous search, there are two main ways of structuring the search space, either on the word histories or on the start time of the words. The first case leads to the popular *Re-Entrant Tree* method strongly associated with the dynamic programming approach where time is the independent variable and the expansion proceeds in parallel in a

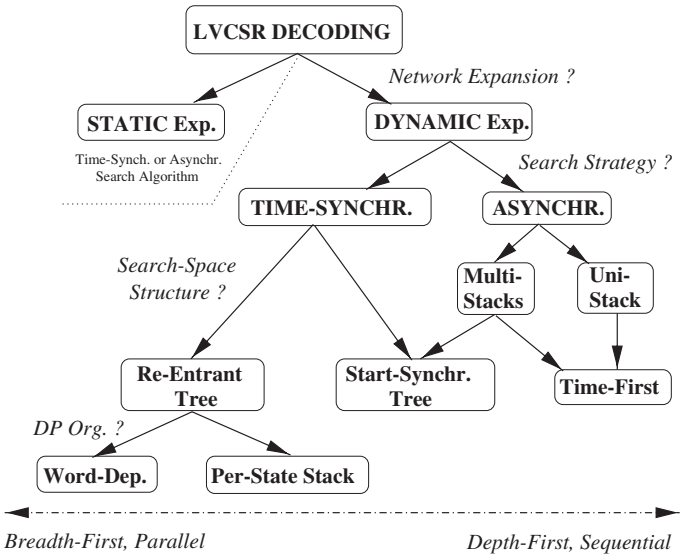

Figure 5. Classification "tree" of decoding techniques

"breadth-first" manner. The second way is an interesting hybrid case that has been considered by both search strategies, namely, the use of start-synchronous trees.

The terms "breadth-first" or "depth-first" are to be interpreted with care in the present context of LVCSR. First, the search is usually expanded at the state level, not at the whole word level and, second, a number of heuristics are always involved to narrow down the network exploration. Time-synchronous search relies heavily on beam pruning to restrict the number of hypotheses developed in parallel and asynchronous stack decoding attempts to implement a "best-first" expansion far beyond a crude "depth-first" search.

5. Static network expansion

Expanding the whole search network prior to decoding is by no means new or recent. Actually, this has long been the most natural approach to LVCSR until the increase of the vocabulary size in conjunction with ever more complex knowledge sources 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 6) or to consider optimization techniques for compacting the network.

5.1. Sparsity of knowledge sources and network redundancies

There are two main sources of potential reduction of the network size:

- exploiting the *sparsity* of the knowledge sources;
- detecting and taking advantage of the *redundancies*.

The sparsity of the current knowledge sources stems essentially from their *regular* structure inherent to simplified model assumptions and partly also from limited training data. The redundancies are, in general, introduced by the systematic *substitution* mechanism used for constructing the network from smaller size units like HMM phones or word models. They are

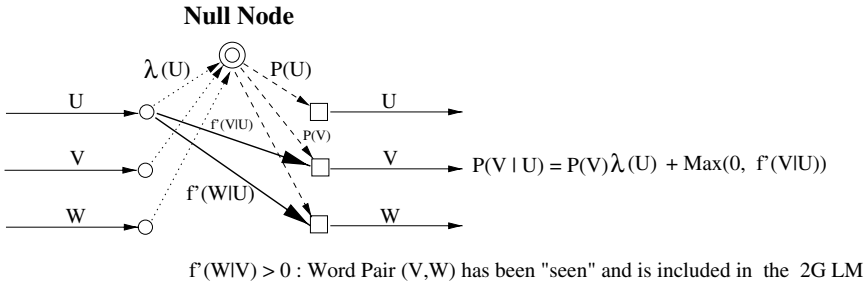

Figure 6. Interpolated backing-off bigram using a null node.

also caused by *ad-hoc* structures like a phonetic prefix-tree where tails are left unshared, and due to model duplication, for example, consecutive to parameter tying. Taking *explicit* advantage of the modelling sparsities and applying optimization techniques drawn from finite-state automata theory are the two principal avenues towards network compactness.

The degree of sparsity is especially striking for an m-gram LM with $m > 2$ but is also an issue with CW position-dependent contexts that are usually "generalized" by decision trees. Let us consider a practical example of a 64 K word trigram, typical of a state-of-the-art LVCSR system. Among the 4 billion possible word bigrams, only five to 15 million are included in the model and, for each of these "seen" word-pair histories, the *average* number of trigrams is comprised between two and five. Such a LM would have about five to 15 million states and 15 to 90 million arcs, requiring between 100 and 600 MB of storage. This means a reduction by seven orders of magnitude with respect to a *plain* 64 K trigram that has about 250 trillion (10^{12}) of arcs.

Concerning CW triphones, the number of distinct models after generalization is typically one order of magnitude smaller than the full inventory of position-dependent contexts. Indeed, there are about $45^3 \times 3 \approx 273\,000$ distinct contexts when distinguishing between word begin, word end and word internal triphones based on 45 monophone symbols while the number of distinct phone HMMs after tying appears to be approximately around 25 000 in a large system.

5.2. Central role of the m-gram LM in the search network

Coming back to the use of an m-gram LM, the probability of a word in an unseen context is generally obtained with an interpolation scheme involving shorter history counts, for example, bigram and unigram counts for a trigram. Along this line, back-off "null" nodes (Placeway, Schwartz, Fung & Nguyen, 1993) have long been used in several systems to take advantage of the small fraction of observed bi- or trigrams. Figure 6 explains the use of a null node in an interpolated backing-off bigram model where $\lambda(U)$ is the backing-off normalization factor and $P(V)$ the unigram prior. When a word-pair has not been "seen" in the training corpus, the bigram probability is factorized in two terms $P(V) \times \lambda(U)$ *without* conditional dependency on the predecessor word U. Consequently, for these unseen word-pairs, recombination can be done similarly to the unigram case, on the null node at each word ending, without the need of so-called word copies.

This backing-off property has been exploited in Antoniol, Brugnara, Cettolo and Frederico (1995) for carrying out a static tree-based representation of a bigram network. The prefix tree of the *whole* lexicon appears only once at the "null" node while the other predecessor nodes

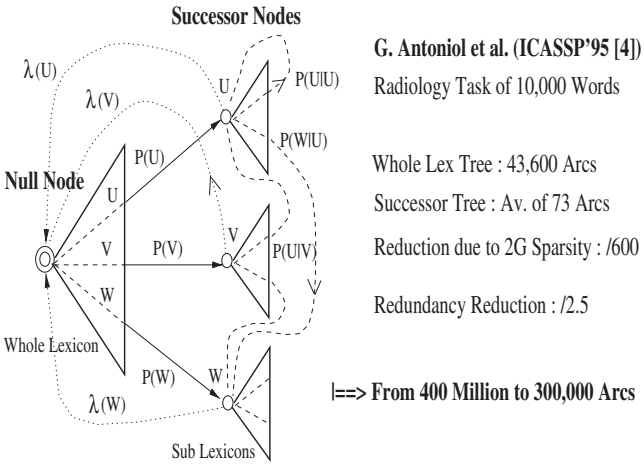

Figure 7. Bigram network with null node and successor trees (Antoniol *et al.*, 1995).

are connected each to a much smaller subtree constructed from the words that belong to the bigrams seen in this linguistic context. As indicated in Fig. 7 below, the average number of phonetic arcs in the subtrees is very small, being reduced by a factor of 600 with respect to the whole tree, due to the characteristics of this radiology task, presumably of low perplexity.

Another advantage of *static* tree-based networks is the ease of factorizing the true *m*-gram probabilities which can be smeared across phone sequences from leaf to root, while building the search network. This results in many linear arc sequences appearing at word endings with an incremental probability of one, consecutive to the factorization process. These linear tails (see Fig. 1) are redundant paths that can be merged and treated in common for all linguistic contexts of that word (Antoniol *et al.*, 1995). Other less trivial redundancies are further reduced by applying general optimization techniques developed in the framework of finite state automata (Aho *et al.*, 1974), leading to another compaction of the network by a factor of two to three. Interestingly, the main impact of these additional redundancy removals is memory saving and *not* speed-up of the decoding (Antoniol *et al.*, 1995), since most of the search effort is spent in the first two tree generations due to the focusing capabilities of beam pruning. This technique has been recently extended to a trigram LM with multiple backing-off nodes as illustrated in Bertoldi, Brugnara, Cettolo, Frederico and Giuliani (2001).

5.3. Weighted finite state transducer method (WFST)

This approach is the outcome of several research years at AT&T and has recently reached the point of becoming an attractive alternative for building a large vocabulary decoder (see a.o. Pereira, Riley & Sproat, 1994; Mohri *et al.*, 1998; Mohri & Riley, 1999; Boulianne, Brousseau, Ouellet & Dumouchel, 2000; Bazzi & Glass, 2000, and elsewhere in this special issue). It offers an elegant unified framework for representing the knowledge sources and producing a search network optimized up to the HMM state level. Along this line, it integrates and extends the main ideas exposed in the previous section devoted to the central role of the LM in the search network. The WFST approach is very 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

all knowledge sources involved in LVCSR like a lexicon with pronunciation variants, stochastic *m*-grams or deterministic phone-in-context expansion rules;

(2) transducers can be combined using the *composition* operator, leading to the integration of the underlying modeling levels in one input-output relation. For example, using the symbol "o" for the composition operator, {C o L o G} would provide a mapping from context-dependent phones up to word sequences, the transducers G, L and C representing, respectively, the grammar model, the context-independent lexicon and the context-dependent phones;

(3) the network is further optimized by weighted determinization followed by minimization, two techniques borrowed from finite state automata theory. An optional step that comes after the network has been determinized, consists of "pushing" the weights towards the initial state much like the already described language smearing technique (Mohri & Riley, 1999). The order in which the individual transducers are composed and optimized might also play a role in obtaining the most compact search network.

The third point deserves some comments concerning the criteria that are pursued for optimizing the network structure. Determinization aims—ideally—at getting a network where any input sequence is matched by, at most, one path, thus reducing the computer time and space for decoding (Mohri, Pereira, Riley & AT&T Labs Research, 2000). In practice, this is a complex task implying a.o. the elimination of all empty arcs such that the total number of arcs might be increased in the determinized network. When applying the WFST method, this pre-processing step is the one that requires the largest computational resources, especially in terms of memory needs. Note that WFST can also be used "on the fly" during decoding and not only for getting a static network expansion. However, this prevents from doing a *global* optimization of the network and makes the decoder more complex.

Some of the main achievements of WFST can be summarized as follows:

- the knowledge sources are handled in a highly flexible way, independently of the decoder specifics, for example, about the contextual scope of the linguistic or phonetic constraints;
- the final optimized network is typically a few times larger than the original LM in terms of number of arcs;
- CW context expansion increases the network by just a few percent with respect to the optimized context-independent network (Mohri *et al.*, 1998).

This last point is quite remarkable and results from postponing the context expansion after having taken advantage of the *m*-gram sparsities and lexical redundancies such that, presumably, relatively few fan-out expansions are still necessary.

Many issues remain open and are currently under study. One direction of work points to the network pre-processing stages and aims at reducing the memory requirements which appear like a bottleneck when very large (language) models are considered. A second direction concerns the best way to handle the final network: should it be loaded in central memory or could it be left on disk and efficiently accessed on-demand? Another related topic concerns the possibility of expanding and optimizing statically some of the knowledge sources while handling the others dynamically. In Demuynck *et al.* (1997), such a "hybrid" approach has been presented where the lexical and phonetic constraints are optimized statically, the LM being decoupled from the other knowledge sources. We will come back to this point in the final conclusion.

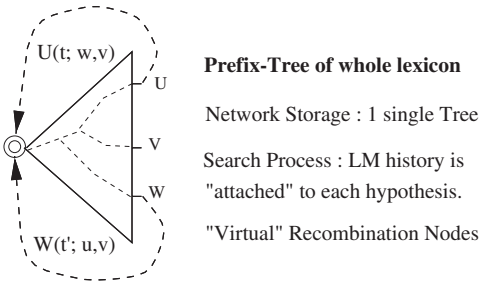

Figure 8. Re-entrant lexical tree for dynamic network expansion.

6. Dynamic search 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 knowledge sources from the outset to get the strongest constraints on the search space has been the key idea leading to one-pass decoders based on dynamically built networks (Jelinek, Bahl & Mercer, 1975; Bahl, Jelinek & Mercer, 1983; Ney, Mergel, Noll & Paeseler, 1987; Ney, Haeb-Umbach, Tran & Oerder, 1992; Odell, Valtchev, Woodland & Young, 1994).

An important aspect has consisted, so far, of assuming the *regularity* of the network structure to deal with the "CW m-gram" constraints, mainly for pragmatic reasons related to algorithmic complexity. 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. It must be understood that this generic tree structure is only stored once and that the search network will be constructed partially and dynamically using virtual nodes and temporary structures containing only the necessary information to process the expanded hypotheses.

Another "key" feature concerns the point of view adopted for structuring the m-gram search space where the emphasis can be placed either on the *linguistic context* or on the *start time* of a word.

Hence, two basic approaches are identified for dynamically exploring an m-gram tree-structured network and generating word sequences of increasing length:

- the **re-entrant tree** where a virtual tree copy is explored for each active linguistic context. This information remains "attached" to each path[6] and recombination is performed at virtual root nodes that depend on the history taken into account by the LM. This method is also known as the "word-conditioned search" (Ney *et al.*, 1992). This is illustrated in Fig. 8;
- the **start-synchronous tree** where a virtual tree copy is being entered at each time requesting successor word hypotheses. All paths having reached a word end hypothesis at a given time are thus extended by exploring the same virtual tree copy associated with this starting time. The terminology has been suggested in Renals and Hochberg (1999), but the method has also been described as the "time-conditioned search" (Ortmanns & Ney, 2000). Figure 9 explains this second strategy.

[6] Similar to the token passing principle (Young, Russel & Thornton, 1989).

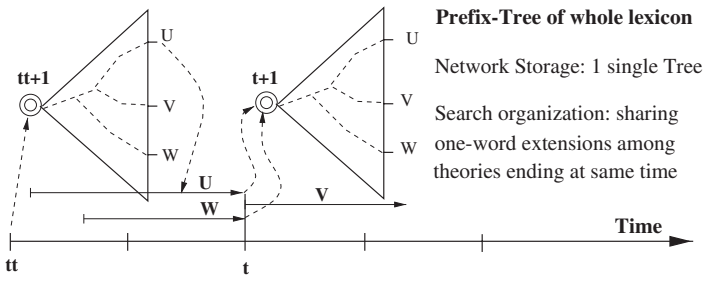

Figure 9. Start-synchronous tree for dynamic network expansion.

Both methods have been applied in the framework of time-synchronous dynamic programming search while the second has also been investigated in multi-stack asynchronous decoders.

6.1. Time-synchronous search based on a re-entrant lexical tree

This appears the most popular technique used for LVCSR, probably because of the conceptual simplicity of the early dynamic programming decoders it is originating from, and because the network expansion closely follows the static counterpart described in Section 5. As in all time-synchronous search methods, state hypotheses are developed in parallel and scored against the same portion of the input signal which leads to a straightforward implementation of beam pruning. The integration of m-gram and CW constraints in this search architecture is, however, non-trivial.

For a correct handling of m-gram probabilities, 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.[7] One important feature here is that the word-boundary optimization is carried out by the DP recurrence at virtual nodes when re-entering the tree (Ortmanns, Ney & Aubert, 1997). This implies that, for each word ending at the current time index, only one segmentation will be selected between that word and its predecessor m–1 history, which is known as the m-tuple optimization (Neukirchen, Aubert & Dolfing, 2000). LM look-ahead is usually performed on-demand for a truncated word history using either a special unigram or bigram model or a cache to store the factorized probabilities (Ortmanns *et al.*, 1996).

Two search organizations have been pursued with a re-entrant tree to fulfil the m-gram optimality criterion and they are best explained with the coordinate system introduced in Section 3.4. The differences mainly concern the *order* in which the three search-space coordinates are spanned, time here being the independent variable as shown in Fig. 10.

6.1.1. The word history conditioned DP organization

The hypotheses active at the "current" time are recorded in lists structured on a three-level hierarchy as shown in Fig. 10. The leading variable is the word history which is coded as an m-gram state index. This means that the three dependent coordinates of the search space are spanned in the following order: LM-State $\rightarrow$ Arc-Id $\rightarrow$ State-Id, the emphasis being on the common predecessor word history shared by the next words being expanded. An

[7] Sub-optimal schemes can also be used when suitable (e.g. Nguyen & Schwartz, 1998) and are addressed in Section 7.

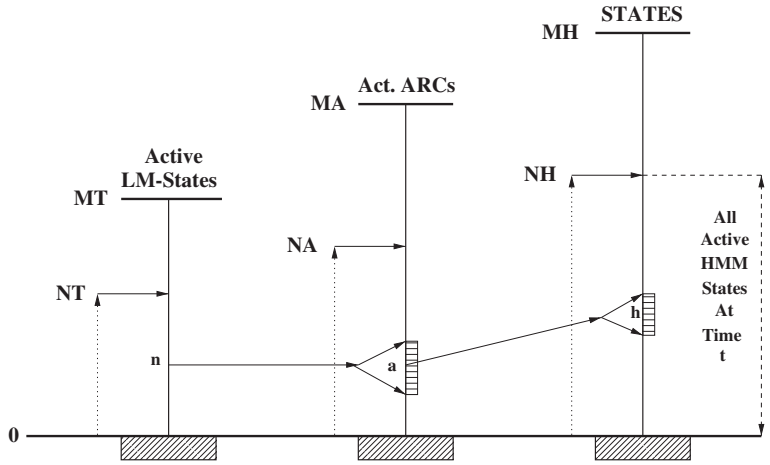

Figure 10. Search organization conditioned on word histories (Ney *et al.*, 1992).

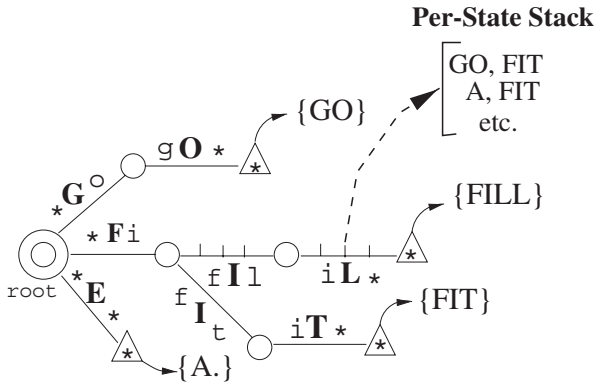

Figure 11. Search organization using the per-state stack (Alleva *et al.*, 1996).

additional structure keeps track of the "back-pointers" to retrieve the preceding words up to the sentence start. This algorithm has been introduced in Ney *et al.* (1992) and further extended in Ortmanns *et al.* (1996) and Aubert (1999).

6.1.2. The per-state stack organization

This method focuses on the multiple instantiations of the same HMM state that happen when a phonetic arc is simultaneously active in different *m*-gram contexts. The ordering of the coordinates becomes: Arc-Id → State-Id → 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 in conjunction with other less "strict" recombination rules (Alleva, 1997; Finke, Fritsch, Koll & Waibel, 1999). Along this line, the concept of subtree dominance leads to a minimax criterion such that all hypotheses contained in a subtree can safely be discarded under certain conditions (Alleva, 1997).

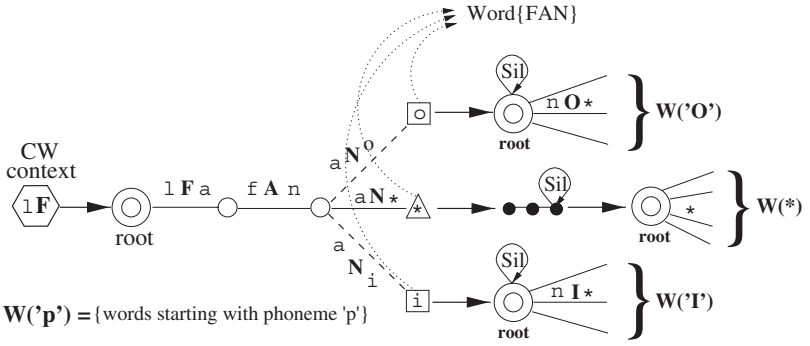

Figure 12. CW transitions with an optional "long" pause.

6.1.3. Integration of CW contexts

In a time-synchronous dynamic search, CW phone modelling implies that multiple contexts have to be considered at word ending to *anticipate* for the next successor words. This leads to the concept of fan-out expansion (Odell *et al.*, 1994) where the last phonetic arc of a word is given several instances, each with a distinct *right* conditioning context. This step can be made more efficient by (1) taking advantage of the redundancy among tied phone HMMs to reduce the fan-out size (Sixtus, Molau, Kanthak, Schlüter & Ney, 2000) and (2) applying a special language look-ahead pruning scheme (Aubert, 1999; Ortmanns, Reichl & Chou, 1999). Indeed, the identity of the right context restricts the set of successor words such that the most promising fan-out arcs can be selected by taking the coming m-gram scores into account. The m-gram recombination at word ends has to take account of this successor constraint (Aubert, 1999) and can possibly be performed *before* expanding the fan-out. Finally, when re-entering the tree, the phonetic arcs have to be selected according to the fan-out right context and the left context of these first generation arcs is specified by the last phone(s) of the previous word. This implies either an "on the fly" instantiation of the corresponding context-dependent HMM or using multiple instances of the first generation arcs. Figure 12 illustrates the main lines of CW transitions in the framework of a re-entrant prefix tree, where two cases of optional pauses are considered between consecutive words: the first one concerns a "short" silence compatible with across-word coarticulation while the second case is a "long" pause that can be followed by any word of the lexicon, as indicated by the wild card symbol $*$ (Aubert, 1999).

6.2. Time-synchronous search based on start-synchronous trees

The main idea here is to share a single one-word extension step among paths ending at the same "current" time, however, with different m-gram 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 the so-called 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 (Oerder & Ney, 1993; Ney, Ortmanns & Lindam, 1997) but makes the word-end recombination step quite expensive (Ortmanns, Ney, Seide & Lindam, 1996). This is, however, compensated by

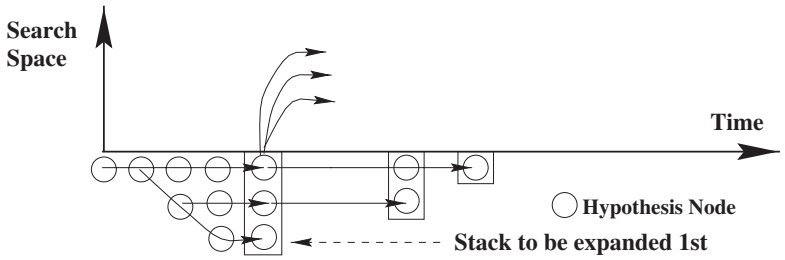

Figure 13. Search expansion in a multi-stack decoder (Schuster, 2000).

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 (Ortmanns *et al.*, 1996; Ortmanns & Ney, 2000).

On the other hand, this architecture appears less favourable for CW context expansions and for m-gram look-ahead pruning schemes involving more than just unigram probabilities. This is a direct consequence of sharing the prefix-tree expansion among simultaneously ending words bearing different linguistic and phonetic (left) contexts (see Fig. 9) as opposed to the re-entrant tree where paths with distinct histories are grouped accordingly. To the best of my knowledge, this method has only been tested with unigram language smearing and for non-cross-word models, so far.

6.3. Asynchronous stack decoding

This approach stems from sequential decoding methods developed some decades ago in communication theory (Jelinek *et al.*, 1975). In LVCSR, a stack decoder 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 (Paul, 1992) or several stacks (Hochberg, Renals, Robinson & Kershaw, 1994), each one grouping the theories associated to the same end-time. Distinguishing between uni- and multi-stack techniques will not be further considered here, being outside the scope of this overview.

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. A good example of such a situation is the computation of N-best sentence hypotheses in a scored word graph where a first backward pass can provide at each node the score of the best path up to the sentence end. For a one-pass decoder, however, an estimation of the future path likelihood can only be obtained by a look-ahead technique. An alternative that does not need looking-ahead in the signal has been presented in Paul (1992) and relies 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 (Gopalakrishnan, Bahl & Mercer, 1995; Renals & Hochberg, 1999; Willett, Neukirchen & Rigoll, 2000) with the consequence that the search space expansion becomes quasi-synchronous i.e. without large end time differences between active theories.

The one-word extensions can either be computed with the start-synchronous tree method of previous section (Hochberg *et al.*, 1994) or using a fast-match algorithm to first get a short list of word candidates that are then processed sequentially for continuing one (or more) theory (Gopalakrishnan *et al.*, 1995; Novak & Picheny, 1999). This fast-match component typically relies on a lexical prefix tree and on simplified acoustic models to achieve the highest efficiency.[8] Concerning the start-synchronous lexical tree exploration, it is worth pointing out that this step has been achieved either with a standard time-synchronous DP scheme (Hochberg *et al.*, 1994) or with a "time-first" asynchronous method (Robinson & Christie, 1998), the latter requiring less memory storage.

Pruning is non-trivial due to the difficulty of comparing the scores of paths having different lengths. The solution consists of 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 (Gopalakrishnan *et al.*, 1995). Based on the current score envelope, a path may be labelled 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 (Gopalakrishnan *et al.*, 1995; Schuster, 2000) by considering left-only conditioned contexts first, and after the one-word extension has been accomplished, proceeding to a re-scoring with the now available right context. The main difference and relative advantage with respect to a time-synchronous search is that CW contexts can be applied on individual word strings with great ease for incorporating longer context ranges (at least to the left), without the need of fan-out expansions. Likewise are long-range LM constraints easily integrated, the recombination step being subjected to the already known dominance principle.

7. Heuristic techniques to further reduce the search space

So far in this paper, beam pruning has been the principal heuristic considered for controlling the size of the explored search space. The heuristic character means this is an *approximation* susceptible of introducing search errors i.e. with no guarantee of finding the best state sequence given the knowledge sources. In this section, two distinct avenues are briefly described for further reducing the search space in addition to the usual beam pruning: the first aims at decoupling the m-gram probabilities from the acoustic–phonetic constraints and the second concerns acoustic look-ahead pruning.

7.1. Decoupling the LM from the acoustic–phonetic constraints

The best state sequence is the one that maximizes the *joint* probability of all knowledge sources given the input signal. Due to the different nature of the acoustic, phonetic and linguistic constraints, decoupling the m-gram contribution is especially appealing as the latter

[8]Time-synchronous search can of course also benefit from such acoustic look-ahead pruning, usually done at the phoneme level as will be seen in Section 7.

involves the longest contexts, namely, word phrases. An interesting avenue is to dissociate as much as possible the optimization of word *time* boundaries from the LM probabilities.

7.1.1. Interactions between LM contribution and word boundaries

The use of long-span LM implies that the hypotheses with distinct word histories have to be kept separate until the m-gram probability can be applied. This means that several instances of the same word model w ending at the same time t, noted as w_t, may occur with different histories $(u, \ldots, v)$ with the consequence that the "optimal" start time τ of w_t is not necessarily unique. It is clear, however, that the segmentation points result from the time-alignment of the acoustic–phonetic models and do not depend directly on the LM. Different start-times τ are likely to occur only when the predecessor words of w_t are *phonetically* distinct and this might be exploited to avoid redundant evaluations of the same word model in the same interval.

As a matter of fact, any continuous-speech decoder has to somehow perform this word boundary optimization step. In the time-synchronous search based on a re-entrant tree, this is implicitly achieved by the recombination at (virtual) tree root nodes and leads to the m-tuple optimization step of the word boundary (Neukirchen *et al.*, 2000):

$$\tau = f(\underbrace{u, \ldots, v, w_t}_{m \text{ words}}; t)$$

which provides a unique start time value of w for each m-tuple ending with w at t. The boundary τ is assumed to depend on the whole m-gram word history and not only, for example, on the phonetic content near the frontier between words v and w. Note that a possible influence on τ of earlier words preceding u is already integrated in the sentence trunk common to all theories being expanded.

With the start-synchronous tree technique, this optimization has to be carried out *explicitly* over all start times of w_t hypotheses that are produced by different "start trees". This in turn implies that these word hypotheses have been stored and can be efficiently retrieved. And similarly for the stack decoding method where this optimization happens sequentially when the successive theories are being expanded. In all cases, there is a potential interest for determining and exploiting the constraints that are really relevant to the word boundaries, especially with the use of LM of still increasing orders, beyond a trigram or a fourgram.

7.1.2. Delayed LM incorporation with heuristic boundary optimization

This search technique appears somewhat hybrid in the sense that it borrows from both the re-entrant tree and start-synchronous tree methods to reach the highest efficiency. The general idea is to assume that the word boundary depends on a narrower context ($<m$ words), possibly phonetically motivated, such that the word expansion and boundary time τ obtained by extending the best theory(ies) can be shared among other m-gram word histories. It is understood that for these alternative phrases the LM will be applied *after* the word expansion has been completed.

This fits with the concept of delayed LM incorporation (Steinbiss, 1991) which has been shown to significantly reduce the active search space by eliminating redundant m-gram word copies. An example of such strategy is given by the word pair approximation (Schwartz & Austin, 1991; Aubert & Ney, 1995) which assumes that the word boundary depends only on

the current and immediate predecessor words i.e.

$$\tau = f(v, w_t; t).$$

LM re-scoring can be subsequently performed with a higher m-gram order ($m > 2$) at very little cost, provided the word hypotheses have been stored in a lattice-like structure based on the word-pair optimized boundaries. As described in Steinbiss (1991) and Seide (2001), this can actually be achieved in one single decoding pass thus taking advantage of the m-gram LM to get a more focused search. In practice, the word-pair approximation has been shown to work quite well but for some very short words like one-phoneme function words (Aubert & Ney, 1995; Ortmanns *et al.*, 1997).

This leads to the idea that the word boundary should actually be made dependent on *phoneme histories* rather than on linguistic word sequences. In Li *et al.* (1996), the estimation of word and phone boundary times has been investigated in the context of a monotone graph search strategy, using either a one-phone or a two-phone look-ahead assumption. Based on U.S.-English experiments, the authors conclude that triphone modelling for speaker-independent recognition can be supported by the two-phone approximation, the one-phone case leading to unacceptable inaccuracies.

A similar approach has been recently pursued in Seide (2001) in the framework of a time-synchronous re-entrant tree decoder, where a *phone-history approximation* is introduced, assuming that the optimal word boundary only depends on the last p phones of a given active path. Experiments have been carried out for Mandarin recognition using half-syllable units and for U.S.-English as well, based on non-cross-word triphones. For LVCSR of Mandarin Chinese, a relative reduction of the search space (in terms of active state hypotheses) in the range of 60–80% has been achieved at no loss of accuracy, which is partly attributed to the structural properties of Chinese as a mono-syllabic language. For U.S.-English on the North American English test-sets, the reduction appears to be in the range of 40–50%, showing clearly the potential offered by appropriate word boundary heuristics to speed-up an m-gram search process.

7.2. Acoustic look-ahead pruning

7.2.1. Principle of a fast acoustic match

The general idea is to pay attention to the acoustic content of the incoming signal (ahead of current hypothesis' time) for *eliminating* as many unpromising candidates as possible, before extension. Provided this preselection is achieved cheaply and reliably, the detailed and expensive knowledge sources can be applied to fewer hypotheses leading to significant speed-ups while maintaining the recognition accuracy. Hence, decoding is achieved in two steps based on some prior "fast match".

Cutting down the search complexity by means of fast match techniques has been considered in the framework of several decoding architectures with rather distinct solutions. In the IBM stack decoder (Novak & Picheny, 1999), a fast acoustic match provides a short list of *word* candidates to extend the most likely theories. Lexical constraints encoded in a prefix tree are used to carry out this fast selection. In the BBN approach Nguyen and Schwartz (1998), relying on the forward-backward search strategy, the fast match component performs a first decoding pass using some linguistic knowledge as well, the results of which are exploited to prune a second detailed pass based on *complete sentence* likelihood estimations. These examples, together with Section 7.2.2, show the wide range of time intervals and knowledge

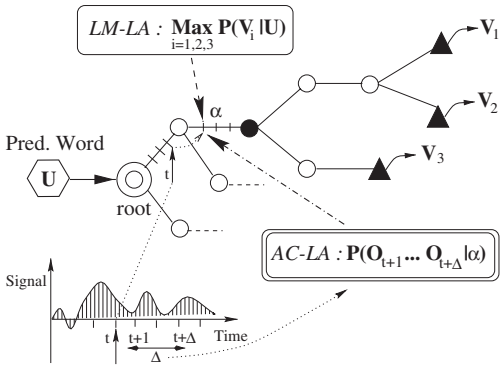

Figure 14. Combined acoustic and LM look-ahead in time-synchronous search (Aubert & Blasig, 2000), LM stands for language models, LA for look-ahead and AC for acoustic model.

sources that are involved in different applications of the fast match principle. All these situations, however, have the use of simplified acoustic models with reduced context dependencies in common.

7.2.2. Phoneme look-ahead in time-synchronous decoders

Phoneme look-ahead pruning has been used for a long time in time-synchronous decoders (Ney *et al.*, 1992). As shown in Fig. 14, it consists of estimating the likelihood of producing a given phone in a short interval ahead of the current time-index, to predict which phonetic arcs in the prefix tree are likely extensions. Given the lexical prefix tree structure, the phonetic transitions are indeed the most convenient place to apply a fast acoustic match possibly working in conjunction with the LM look-ahead scores (Aubert & Blasig, 2000). However, working at the phone level clearly limits the selection capabilities since only short-range lexical or linguistic constraints can be taken into account and the look-ahead time interval must be kept relatively short.

The phonetic look-ahead scores are usually computed with coarse acoustic models as well as with simplified time-alignments (Ortmanns *et al.*, 1996; Alleva, 1997) such that the admissibility of this pruning cannot be guaranteed. In Aubert and Blasig (2000), such a technique has been evaluated for broadcast news transcriptions and the search space could be reduced by a factor of more than two at almost no degradation. Although there is an obvious tradeoff between speed and accuracy, this reduction appears quite appreciable just by looking ahead of one phonetic arc and less than one-tenth of a second of speech.

8. Some experimental evidence in large vocabulary decoders

The problem of evaluating and comparing different decoding algorithms is addressed by first discussing which experimental methodology is the most suitable and by looking at the results achieved by three state-of-the-art large-vocabulary recognition systems during the DARPA evaluation held in December 1999.

8.1. Specification and evaluation of a "real" decoder

When implementing a practical large-vocabulary decoder, the challenge is—obviously—to get the highest accuracy from the knowledge sources at the lowest costs in terms of memory space and computational time.

As already mentioned, the potential size of the search space prevents from applying any "brute force" method (exhaustive search) and requires the use of complex algorithms in conjunction with powerful heuristics including the usual beam pruning. Consequently, for any LVCSR task, finding the *optimal* state sequence cannot be guaranteed and there is always a tradeoff between accuracy and computational costs. Note, however, that the so-called search errors normally represent a very small percentage ($\approx$1 or 2%) of all errors that are mainly caused by using imperfect models and knowledge sources.

For evaluating a given decoder, it is important to consider several working conditions i.e. distinct degrees of pruning and not just one fixed setup. If, for example, the baseline system has been run with conservatively large pruning thresholds, any search improvement is likely to provide a dramatic speed up effect! Along this line, it is extremely useful to establish the characteristic curve of a decoder, showing the recognition accuracy as a function of the computational resources over a *wide range* of operating conditions. Typically, this leads to a curve showing the word error rate against the real-time factor and the slope of this curve gives some indication about the efficacy of the pruning strategies. In Gauvain and Lamel (2000), such an evaluation has been conducted on broadcast news data, both for single and multiple pass decoding. The influence of the acoustic model size and of the LM order has also been investigated regarding the performance that can be achieved under restricted computational resources.

8.2. Results of Hub-4 10$\times$ real-time systems

Within the framework of DARPA evaluations, "Hub-4" stands for automatic transcriptions of broadcast news recordings as they are found in the real radio or television media world. For one optional test case, the participants have been requested to run their system on a single processor below the 10$\times$ real-time limit

Table I summarizes the main lines of the results produced in December 1999 by three groups having contributed to this 10$\times$ real-time "spoke". It has to be emphasized that these systems are quite elaborate *multi-pass* decoders including unsupervised acoustic adaptation and higher order LM re-scoring. The first pass however is the most CPU demanding and here too, the three decoders considered in the table are quite different, being representative of three distinct classes of methods. The first two systems include a fast match algorithm while the third one rests upon a standard time-synchronous dynamic-expansion decoder based on the re-entrant tree method. In all three cases the vocabulary size is about 64 K words.

Referring to Table I, the following comments can be made:

- the final word error rates are quite similar differing by only $\pm$2% relative;
- some small differences in memory needs may be observed with a slight advantage for the asynchronous "time-first" approach.

Looking at these results, it may be argued that there is no evidence of a clearly "dominant" decoding method. Indeed, it might have been expected that the constraint imposed on the real-time factor would have led to larger differences in accuracy which has not been the case, so far. But the main conclusion, admittedly, is that each site has been successful at developing a multiple pass system based on their own algorithmic expertise.

TABLE I. DARPA Dec'99 Hub-4 results for 10× real-time broadcast news systems (DARPA, 2000) FM stands for fast (acoustic) match and WER for word error rate in %

Group	1st-pass algorithm	Hardware specifics	WER % final
BBN	Fw-Bw Search Time-Sync FM	Pentium III 600 MHz <1 GB RAM	17·3%
IBM	Asynchronous Stack + FM	RS/6000, 320 Mips 512 MB RAM	17·6%
LIMSI	Time-Synchr. Dyn. Network	Compaq XP 500 MHz Peak 800 MB	17·1%

8.3. The successful "trilogy" in automatic speech recognition

The explanation for not observing significant performance departures among quite different decoders may be found in the simple observation that large vocabulary decoding is a complex problem that cannot be solved with a single or unique algorithm. Given the many interdependencies existing in a large vocabulary continuous speech system, it seems that the best performing systems are almost always the results of a so-called *successful trilogy*, namely, that they are the careful outcome of the following contributions:

1. a number of powerful algorithms (not just one!);
2. a clever design and implementation cooperative with the hardware;
3. a careful tuning of all heuristics (pruning thresholds, penalties, scaling factors).

9. Conclusion

In attempting to summarize this short tour of decoding techniques, the following pros and cons can be suggested:

(1) static network expansion using WFST appears quite general, leads to a very flexible decoding environment and integrates CW phonetic contexts as well as grammar spreading with almost no overhead costs. However, the feasibility of this approach relies strongly on current model sparsities which might be a handicap, especially regarding the LM;
(2) time-synchronous dynamic search achieves very efficient pruning and recombination in a conceptually simple and unified framework. The handling of LM word copies has proven surprisingly efficient. Integrating CW models is however challenging if not tricky and does not seem to "scale" easily to larger phonetic contexts. This method 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 favourable pruning conditions, may be more influenced by fast match performance. However, they offer an ideal decoupling with respect to the LM interface and are well-suited for integrating longer contextual constraints of whatever nature.

Finally, without guessing at any kind of predictions about what could be tomorrow's decoders, here are a number of avenues that are currently being studied and appear definitely worth pursuing:

- *hybrid expansion strategies*: combining a static expansion of certain knowledge sources with an "on the fly" dynamic integration of the others is attractive since the acoustic–phonetic constraints are the most cumbersome to implement efficiently and the language constraints are the most memory intensive. However, other combinations are also being considered to take advantage of the massive m-gram sparsities;
- *increasing importance of word-graphs*: given the interest in proposing numerous sentence alternatives and the growing importance of confidence measures related to word posterior probabilities, the backbone of a LVCS decoder could become a word-graph data structure, besides replacing the necessary back-pointers to retrieve whole sentence hypotheses;
- *integration of very long range syntactical constraints*: given the limitations of current m-gram LM, a good deal of work is being spent on more syntactical approaches implying much longer contextual dependencies, beyond trigram or fourgram. These in turn will stimulate new search heuristics to benefit from the LM predictive power in the earliest decoding stage.

I would like to thank my colleagues at Philips Research in Aachen for their support, especially Christoph Neukirchen, Hans Dolfing and Matthew Harris, with whom I had many enlightening discussions while preparing this overview. It is also my pleasure to acknowledge the contribution of Frank Seide (Philips Research East Asia) to the content of Section 7.

References

Aho, A. V., Hopcroft, J. E. & Ullman, J. D. (1974). *The Design and Analysis of Computer Algorithms*. Addison Wesley.

Alleva, F., Huang, X. & Hwang, M.-Y. (1996). Improvements on the pronunciation prefix tree search organization. *Proceedings of the International Conference on Acoustics, Speech and Signal Processing*, Atlanta, GA, pp. 133–136.

Alleva, F. (1997). Search organization in the whisper continuous speech recognition system. *Proceedings of the IEEE ASRU Workshop*, Santa Barbara, CA, pp. 295–302

Antoniol, G., Brugnara, F., Cettolo, M. & Frederico, M. (1995). Language model representations for beam-search decoding. *Proceedings of the International Conference on Acoustics, Speech and Signal Processing*, Detroit, MI, pp. 588–591.

Aubert, X. & Ney, H. (1995). Large vocabulary continuous speech recognition using word graphs. *Proceedings of the International Conference on Acoustics, Speech and Signal Processing*, Detroit, MI, USA, pp. 49–52.

Aubert, X. (1999). One pass cross word decoding for large vocabularies based on a lexical tree search organization. *Proceedings of the European Conference on Speech Communication and Technology*, Budapest, Hungary, pp. 1559–1562.

Aubert, X. & Blasig, R. (2000). Combined acoustic and linguistic look-ahead for one-pass time-synchronous decoding. *Proceedings of the International Conference on Spoken Language Processing*, volume 3, Bejing, China, pp. 802–805.

Bahl, L. R., Jelinek, F. & Mercer, R. L. (1983). A maximum likelihood approach to continuous speech recognition. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, volume 5, pp. 179–190.

Bazzi, I. & Glass, J. (2000). Heterogeneous lexical units for automatic speech recognition: preliminary investigations. *Proceedings of the International Conference on Acoustics, Speech and Signal Processing*, Istanbul, Turkey, pp. 1257–1260.

Bellman, R. E. (1957). *Dynamic Programming*. Princeton University Press, Princeton, NJ.

Bertoldi, N., Brugnara, F., Cettolo, M., Frederico, M. & Giuliani, D. (2001). From broadcast news to spontaneous dialogue transcription: portability issues. *Proceedings of the International Conference on Acoustics, Speech and Signal Processing*, Salt Lake City, Utah, USA, pp. 37–40.

Boulianne, G., Brousseau, J., Ouellet, P. & Dumouchel, P. (2000). French large vocabulary recognition with cross-word phonology transducers. *Proceedings of the International Conference on Acoustics, Speech and Signal Processing*, Istanbul, Turkey, pp. 1675–1678.

DARPA (2000). Dec'99 evaluation of 10× real-time transcriptions of broadcast news. *DARPA Proceedings of the Broadcast News Workshop*, Morgan Kaufmann Publishers, San Francisco.

Demuynck, K., Duchateau, J. & Van Compernolle, D. (1997). A static lexicon network representation for cross-word context-dependent phones. *Proceedings of the European Conference on Speech Communication and Technology*, volume 1, Rhodos, Greece, pp. 143–146.

Finke, M., Fritsch, J., Koll, D. & Waibel, A. (1999). Modeling and efficient decoding of large vocabulary conversational speech. *Proceedings of the European Conference on Speech Communication and Technology*, volume 1, Budapest, Hungary, pp. 467–470.

Gauvain, J. L. & Lamel, L. (2000). Fast decoding for indexation of broadcast data. *Proceedings of the International Conference on Spoken Language Processing*, volume 3, Bejing, China, pp. 794–797.

Gopalakrishnan, P. S., Bahl, L. R. & Mercer, R. L. (1995). A tree search strategy for large-vocabulary continuous speech recognition. *Proceedings of the International Conference on Acoustics, Speech and Signal Processing*, Detroit, WA, pp. 572–575.

Hanazawa, K., Yasuhiro, M. & Furui, S. (1997). An efficient search method for large-vocabulary continuous-speech recognition. *Proceedings of the International Conference on Acoustics, Speech and Signal Processing*, volume 3, Munich, Germany, pp. 1787–1790.

Hochberg, M., Renals, S., Robinson, A. & Kershaw, D. (1994). Large vocabulary continuous speech recognition using a hybrid connectionist-HMM system. *Proceedings of the International Conference on Spoken Language Processing*, Yokohama, Japan, pp. 1499–1502.

Jelinek, F., Bahl, L. R. & Mercer, R. L. (1975). Design of a linguistic statistical decoder for the recognition of continuous speech. *IEEE Transactions on Information Theory*, **IT-21**, 250–256.

Li, Z., Boulianne, G., Labute, P., Barszcz, M., Garudadri, H. & Kenny, P. (1996). Bi-directional graph search strategies for speech recognition. *Computer Speech and Language*, **10**, 295–321.

Mohri, M., Riley, M., Hindle, D., Ljolje, A. & Pereira, F. (1998). Full expansion of context-dependent networks in large vocabulary speech recognition. *Proceedings of the International Conference on Acoustics, Speech and Signal Processing*, WA, Seattle, pp. 665–668.

Mohri, M. & Riley, M. (1999). Network optimizations for large-vocabulary speech recognition. *Speech Communication Journal*, **28**, 1–12.

Mohri, M., Pereira, F., Riley, M. & AT&T Labs Research,, (2000). Weighted finite-state transducers in speech recognition. *ASR'2000 Proceedings*, Paris, France, pp. 97–106.

Neukirchen, C., Aubert, X. & Dolfing, H. (2000). Extending the generation of word graphs for a cross-word M-gram decoder. *Proceedings of the International Conference on Spoken Language Processing*, volume 4, Bejing, China, pp. 302–305.

Ney, H., Mergel, D., Noll, A. & Paeseler, A. (1987). A data driven organization of the dynamic programming beam search for continuous speech recognition. *Proceedings of the International Conference on Acoustics, Speech and Signal Processing*, Dallas, Texas, pp. 833–836. (1992) Also in *IEEE Transaction on Signal Processing*, **SP-40**, 272–281.

Ney, H., Haeb-Umbach, R., Tran, B. H. & Oerder, M. (1992). Improvements in beam search for 10000-word continuous speech recognition, pp. 13–16. *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, volume I, San Francisco, CA, pp. 9–12.

Ney, H., Ortmanns, S. & Lindam, I. (1997). Extensions to the word graph method for large vocabulary continuous speech recognition. *Proceedings of the International Conference on Acoustics, Speech and Signal Processing*, Munich, pp. 1787–1790.

Nguyen, L. & Schwartz, R. (1998). The BBN single-phonetic-tree fast-match algorithm. *Proceedings of the International Conference on Spoken Language Processing*, Sydney, Australia, pp. 1827–1830.

Nilsson, N. J. (1998). *Artificial Intelligence: A New Synthesis*. Morgan Kaufmann Publishers, Inc., San Fransisco, CA.

Novak, M. & Picheny, M. (1999). Speed improvement of the time-asynchronous acoustic fast match. *Proceedings of the European Conference on Speech Communication Technology '99*, Budapest, Hungary, pp. 1115–1118.

Odell, J. J., Valtchev, V., Woodland, P. C. & Young, S. J. (1994). A one pass decoder design for large vocabulary recognition. *Proceedings of the ARPA Spoken Language Technology Workshop*, Plainsboro, NJ, pp. 405–410.

Oerder, M. & Ney, H. (1993). Word graphs: an efficient interface between continuous speech recognition and language understanding. *Proceedings of the International Conference on Acoustics, Speech and Signal Processing*, volume 2, Minneapolis, MN, pp. 119–122.

Ortmanns, S., Ney, H. & Eiden, A. (1996). Language-model look-ahead for large vocabulary speech recognition. *Proceedings of the International Conference on Spoken Language Processing 96*, Philadelphia, pp. 2095–2098.

Ortmanns, S., Ney, H., Seide, F. & Lindam, I. (1996). A comparison of time conditioned and word conditioned search techniques for large vocabulary speech recognition. *Proceedings of the International Conference on Spoken Language Processing,* Philadelphia, pp. 2091–2094.

Ortmanns, S., Ney, H. & Aubert, X. (1997). A word graph algorithm for large vocabulary continuous speech recognition. *Computer Speech and Language*, **11**, 43–72.

Ortmanns, S., Firzlaff, T. & Ney, H. (1997). Fast likelihood computation methods for continuous mixture densities in large vocabulary speech recognition. *Proceedings of the European Conference on SpeechCommunication and Technology '97,* Rhodes, Greece, pp. 139–142.

Ortmanns, S., Reichl, W. & Chou, W. (1999). An efficient decoding method for real-time speech recognition. *Proceedings of the European Conference on Speech Communication and Technology,* Budapest, Hungary, pp. 499–502.

Ortmanns, S. & Ney, H. (2000). The time-conditioned approach in dynamic programming search for LVCSR. *IEEE Transaction of Speech and Audio Processing*, **8**, 676–687.

Paul, D. B. (1992). An efficient A^* decoder algorithm for continuous speech recognition with a stochastic language model. *Proceedings of the International Conference on Acoustics, Speech and Signal Processing,* volume 1, CA, pp. 25–28.

Pereira, F., Riley, M. & Sproat, R. (1994). Weighted rational transductions and their application to human language processing. *Proceedings of the ARPA HLT Workshop*.

Placeway, P., Schwartz, R., Fung, P. & Nguyen, L. (1993). The estimation of powerful language models from small and large corpora. *Proceedings of the International Conference on Acoustics, Speech and Signal Processing,* volume II, Minneapolis, MN, USA, pp. 33–36.

Renals, S. & Hochberg, M. (1999). Start-synchronous search for large vocabulary continuous speech recognition. *IEEE Transactions on Speech and Audio Processing*, **7**, 542–553.

Robinson, T. & Christie, J. (1998). Time-first search for large vocabulary speech recognition. *Proceedings of the International Conference on Acoustics, Speech and Signal Processing,* Seattle, USA, pp. 829–832.

Schramm, H. & Aubert, X. (2000). Efficient integration of multiple pronunciations in a large vocabulary decoder. *Proceedings of the International Conference on Acoustics, Speech and Signal Processing,* volume 3, Istanbul, Turkey, pp. 1659–1662.

Schuster, M. (2000). Memory-efficient LVCSR search using a one-pass stack decoder. *Computer Speech and Language*, **14**, pp. 47–77.

Schwartz, R. & Austin, S. (1991). A comparison of several approximate algorithms for finding multiple (N-BEST) sentence hypotheses. *Proceedings of the International Conference on Acoustics, Speech and Signal Processing,* Toronto, Canada, pp. 701–704.

Seide, F. (2001). The use of virtual hypothesis copies in decoding of large-vocabulary continuous speech. Philips Research East-Asia, submitted to IEEE Transactions on Speech and Audio Processing, Taipei.

Sixtus, A., Molau, S., Kanthak, S., Schlüter, R. & Ney, H. (2000). Recent improvements of the RWTH large vocabulary speech recognition system on spontaneous speech. *Proceedings of the International Conference on Acoustics, Speech and Signal Processing,* Istanbul, Turkey, pp. 1671–1674.

Steinbiss, V. (1991). A search organization for large vocabulary recognition based on N-best decoding. *Proceedings of the European Conference on Speech Communication and Technology,* Genova, Italy, pp. 1217–1220.

Steinbiss, V., Tran, B. H. & Ney, H. (1994). Improvements in beam search. *Proceedings of the International Conference on Spoken Language Processing,* Yokohama, Japan, pp. 2143–2146.

Willett, D., Neukirchen, C. & Rigoll, G. (2000). Ducoder—the Duisburg University LVCSR stack decoder. *Proceedings of the International Conference on Acoustics, Speech and Signal Processing,* Istanbul, Turkey, pp. 1555–1558.

Young, S. J., Russel, N. H. & Thornton, J. H. (1989). Token passing: a simple conceptual model for connected speech recognition systems. Technical Report F_INFENG/TR38, Cambridge University Engineering Department.

(Received 18 July 2001 and accepted for publication 12 October 2001)