status report for

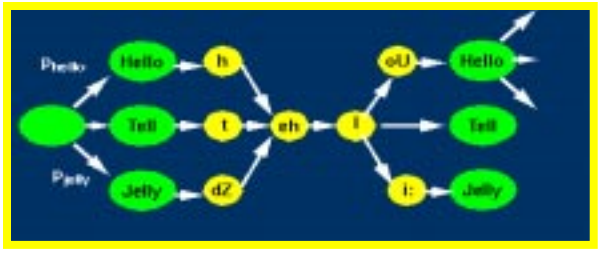
Fast Recognition Techniques for Large Vocabulary Recognition

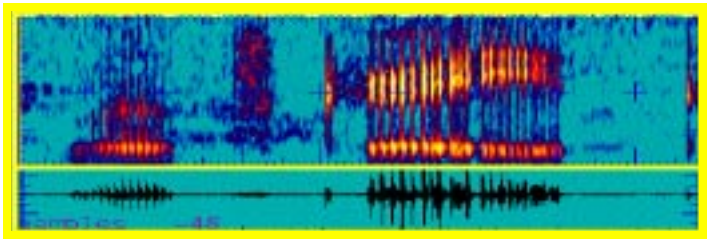
submitted to:

Dr. Yu-Hung Kao
Personal Systems Laboratory
Texas Instruments, Inc.
MS 8374, PO Box 655303
Dallas, Texas 75265
Email: yhkao@csc.ti.com

December 7, 1998

TEXAS INSTRUMENTS THE WORLD LEADER IN DIGITAL SIGNAL PROCESSING SOLUTIONS





submitted by:

J. Zhao, J. Hamaker, N. Deshmukh, A. Ganapathiraju, J. Picone
Institute for Signal and Information Processing
Department of Electrical and Computer Engineering
Mississippi State University
Box 9571
413 Simrall, Hardy Road
Mississippi State, Mississippi 39762
Tel: 601-325-3149, Fax: 601-325-3149
Email: {zhao, picone}@isip.msstate.edu

A. INTRODUCTION

In recent years we have seen great advances in speech recognition technology. Large vocabulary systems are now finding their way into the marketplace, mainly in the desktop and server markets. Typically these systems are restricted to a particular domain such as automatic dictation or command-and-control applications. With these restrictions, developers are able to create highly efficient systems which run in real-time with very low error rates. However, the primary goal of speech research is to produce systems that allow users to interact naturally without restrictions on either content or style of speech. Unfortunately, the resources required for a state-of-the-art conversational speech recognition system to be commercially viable in the handheld computing market stretch far beyond available hardware technology.

The majority of this resource consumption is owed to the search process which combines multiple knowledge sources to find the word path which best matches the available statistical models of speech. The Institute for Signal and Information Processing (ISIP) is developing a public domain speech recognition system which simultaneously supports various search algorithms optimized for different applications, vocabulary sizes, etc. [1-3]. In conjunction with this project, ISIP is working with Texas Instruments (TI) to investigate algorithms which limit the resources used by the search process while maintaining the accuracy of the search [4]. The ultimate goal of this work is to provide search algorithms which will decode LVCSR grammars in real-time on a TI DSP platform.

State-of-the-art LVCSR decoders have become very complex and resource-intensive. For some applications decoding a simple one-word sentence may take tens of Megabytes of memory and may run at 100 times real-time. This is due to the overhead required to employ multiple layers of knowledge such as the N-gram language models, detailed acoustic models (triphones), etc. It is our belief that to create an extremely efficient decoder one must either abandon the use of this high-level knowledge or find ways to make the application of those knowledge sources more efficient. Of course, the former is not an option. Thus, ISIP continues to investigate algorithms and techniques which will make each part of the decoder as efficient as possible. These extremely efficient components will then be combined to create an efficient overall search. In this report we describe two of the issues we have investigated to this point: acoustic look-ahead and efficient N-gram decoding.

B. ACOUSTIC LOOK-AHEAD

Most state-of-the-art speech recognition systems use pruning techniques to reduce the search space. In these, a threshold is set at each level in the search where only paths whose score falls above that threshold are extended to the next level, pruning away all others. We examined a two-pass fast-match decoding strategy [5] where the first pass quickly finds an approximate solution by applying a simple heuristic at each level of the search. The second pass uses the knowledge gained from the first pass to perform a more detailed search. For a real-time system, a full two-pass decoding where the decoder waits for all of the input speech before decoding is not practical. However, we believe that this technique can be extended to a short-time look-ahead similar to language model look-ahead [6,7] and stack decoding [8,9]. The art to this type of algorithm is determining the heuristic which can find a high quality partial solution using very limited resources.

As a first test of this technology we implemented a heuristic which extends only the N-best paths at each point through the search. At the end of this first pass, a best overall hypothesis is found and the partial path score at every level of the hypothesis is then used as the threshold for the second pass of the search. We compared this algorithm against Viterbi with beam pruning on a small subset of the OGI Alphadigits Corpus [10] and found that the fast-match search produced a much tighter beam with little effect on WER. The fast-match scheme also gave a second pass which was significantly faster than Viterbi with beam pruning. However, the overhead required to find the thresholds in the first pass was substantial as demonstrated in Figure 1.

This work is currently being extended to include heuristics which provide a more efficient overall search process. An example which has given encouraging preliminary results is the use of simple monophone acoustic models for the look-ahead pass of the search and context-dependent triphone models for the second pass. We are also verifying our initial Alphadigits results on the far more complex SWITCHBOARD task [11].

C. EFFICIENT N-GRAM DECODING

Large vocabulary continuous speech recognition requires the use of a language model or grammar to select the most likely word sequence from the relatively large number of alternative word hypotheses produced during the search process. A language model that uses the history of the $n-1$ immediately preceding words to compute the occurrence probability P of the current word is called an N-gram language model. The value of N is typically limited to 2 (bigram model) or 3 (trigram model) for feasibility. Obviously, it is not possible for an N-gram language model to estimate probabilities for all possible word pairs. Typically an N-gram lists only the most

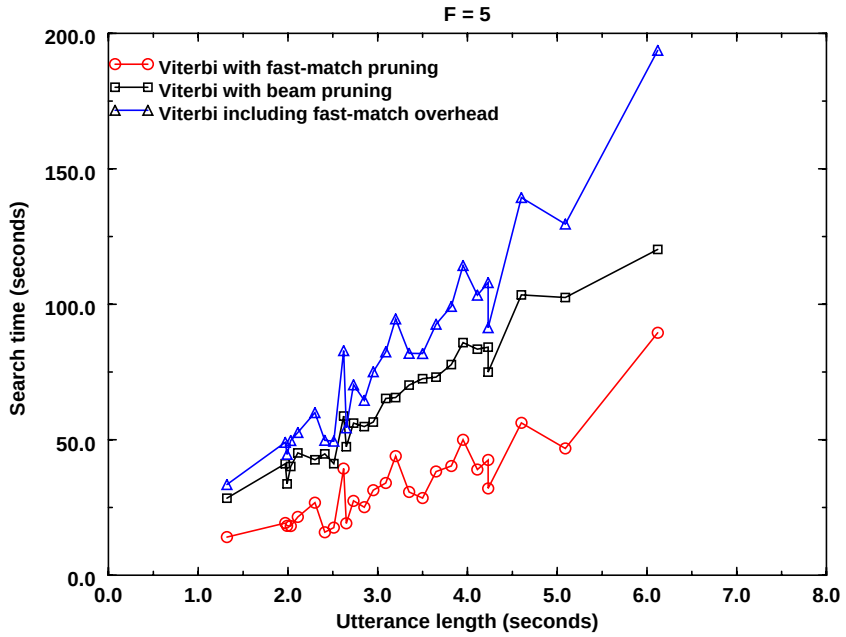


Figure 1. Comparison of search efficiency when using the fast-match look-ahead and efficient beam pruning. The fast-match look-ahead produces a more restrictive beam than the beam pruning technique but the overhead required is excessive. Clearly a more efficient look-ahead algorithm could improve the efficiency and quality of the search.

frequently occurring word pairs, and uses a backoff mechanism to compute the probability when the desired word pair is not found. For example, in the case of a bigram, if $P(W_i, W_j)$ is sought and is not found, the probability is computed as $b(\dot{W}_i) \cdot p(W_j)$, where $b(W_i)$ is the back-off score for W_i. Other higher-order backoff N-gram language models can be defined similarly [8].

In order to efficiently store the N-gram language model and compute the LM scores, we construct two special data structures — N-gram and N-gram node. We use a hash-table mechanism to store these N-gram nodes for quick access during decoding. Typically, when a word end is reached during the decoding process, the decoder constructs a new lexical tree encompassing all the possible next words which are used to generate the next phone hypotheses. The scores for word transitions are stored in the lexical tree and computed based on the position of the word in this lexical tree and the current N-gram history. In N-gram decoding, each word (except the sentence start word) can be followed by any other word, so for large vocabulary speech recognition the lexical tree would be very large.

It is a memory intensive process to create an N-gram lexical tree for each N-gram word. However, we know that the N-gram lexical tree structure is the same for all occurrences of each word, only the LM scores vary according to their N-gram histories. Hence, instead of making several copies of the lexical tree, we use the same N-gram lexical tree for each occurrence of a word, and compute the LM scores (including backoff LM scores) on the fly while decoding. Thus memory requirements are significantly minimized, and we can achieve efficient N-gram decoding [12].

There is much more work to be done if this technology is to be applied to a DSP platform. As a next step, we plan to use a more efficient LM storage mechanism that parallels cache usage in most modern processors [13]. At any point in time we envision about 10% of the LM in memory and the rest on disk (or in ROM in the case of a DSP platform). Replacement techniques like Least Recently Used or Random replacement will be attempted. This approach should reduce memory usage considerably without a significant compute time overhead.

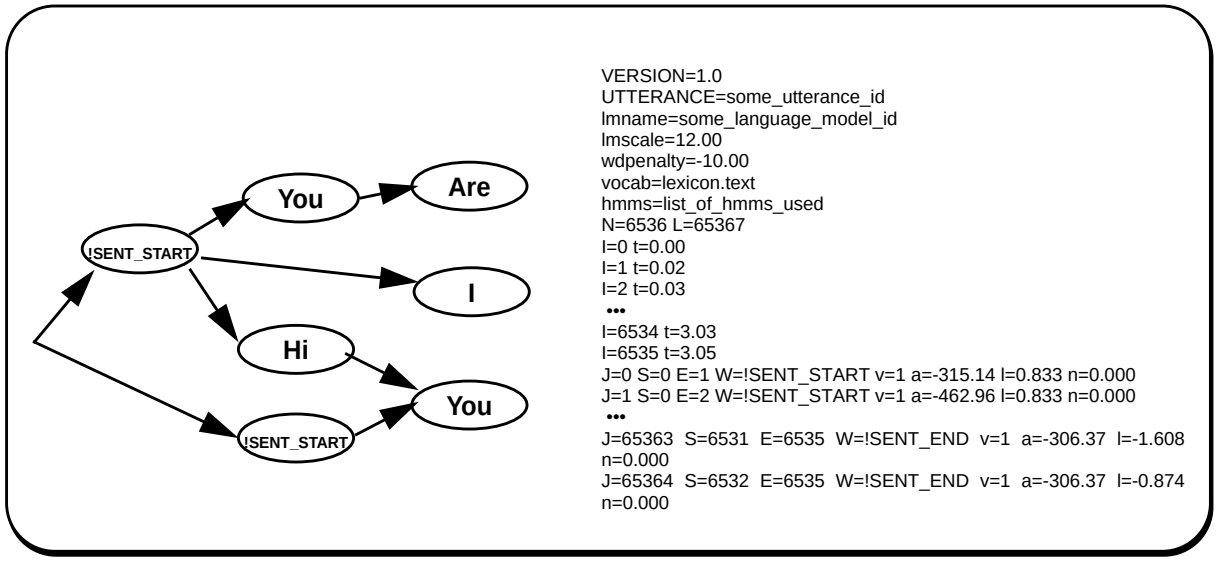


Figure 2. Word graph generation using an N-gram language model.

To supplement the N-gram decoding, we have also implemented grammar compilation and N-gram word-graph generation. A word graph is a point-to-point description of the search through the language model as indicated in Figure 2. Once the word graph is generated, it can be applied in a second pass decoding (typically referred to as lattice rescoring), in which the search space is greatly reduced in comparison to using the N-gram language model. Also, this word graph can be reused to efficiently test new acoustic models or language models at a reduced computational complexity.

In N-gram word-graph generation, the decoder keeps an N-best list of hypotheses instead of only the 1-best at each level (word, acoustic model, state). We have developed a very efficient method for compiling N-best lists into a word graph. For generating a word graph, we modified the ISIP decoder by allowing each path marker object to maintain a list of backpointers instead of a single backpointer. When two paths need to be merged without destroying either of them, we simply group their respective backpointers together and assign them to the better of the two paths, and thereafter grow only the better path. Thus the path histories are maintained at no significant additional cost in memory.

SUMMARY

The fast recognition algorithms we have implemented have been found to effectively reduce the search space, and improve the decoder speed. The progress on this work to date has been encouraging but there is much work to be done before large vocabulary tasks can be performed in real-time on a DSP platform. Much of this work will involve developing efficient structures to hold the complex language models associated with continuous speech recognition. Also, we must continue to refine the search algorithms. The work presented in this report is currently being extended to search heuristics which provide a more efficient overall search process. An example which has given encouraging preliminary results is the use of simple monophone acoustic models for an acoustic look-ahead and context-dependent triphone models for the second pass. All of the software for this work is implemented in C++ using public domain GNU compiler and is available at *http://www.isip.msstate.edu/resources/technology/projects/1998/speech_recognition/*.

REFERENCES

[1] N. Deshmukh, A. Ganapathiraju, J. Hamaker and J. Picone, “An Efficient Public Domain LVCSR Decoder”, Institute for Signal and Information Processing, Mississippi State University, August 1998.

[2] N. Deshmukh, A. Ganapathiraju, J. Hamaker and J. Picone, “Large Vocabulary Conversational Speech Recognition”, *http://www.isip.msstate.edu/resources/technology/projects/current/speech_recognition*, Institute for Signal and Information Processing, Mississippi State University, 1998.

[3] N. Deshmukh, A. Ganapathiraju, J. Hamaker and J. Picone, “An Efficient Public Domain LVCSR Decoder”, in *Proceedings of the Hub-5 Conversational Speech Recognition (LVCSR) Workshop*, Linthicum Heights, Maryland, USA, September 1998.

[4] J. Picone, “Fast Recognition Techniques for Large Vocabulary Recognition”, Institute for Signal and Information Processing, Mississippi State University, December 1997.

[5] J. Zhao, J. Hamaker, N. Deshmukh, A. Ganapathiraju and J. Picone, “Fast Search Algorithms for Continuous Speech Recognition”, submitted to the *IEEE Southeastcon*, Lexington, Kentucky, USA, March 1999.

[6] S. Ortmanns, H. Ney and A. Eiden, “Language Model Look-ahead for Large Vocabulary Speech Recognition”, in *Proceedings of the Fourth International Conference on Spoken Language Processing*, pp. 2095-2098, Philadelphia, Pennsylvania, USA, October 1996.

[7] L. Nguyen, R. Schwartz, F. Kubala and P. Placeway, “Search Algorithms for Software-Only Real-Time Recognition with Very Large Vocabularies”, in *Proceedings of the DARPA Human Language Technology Workshop,* pp. 91-95, Cambridge, Massachusetts, USA, March 1993.

[8] M. Ravishankar, “Efficient Algorithms for Speech Recognition”, Ph.D. Thesis, Carnegie Mellon University, Pittsburgh, Pennsylvania, USA, 1996.

[9] H. Murveit, J. Butzberger, V. Digalakis and M. Weintraub, “Progressive-Search Algorithms for Large Vocabulary Speech Recognition”, in *Proceedings of the DARPA Human Language Technology Workshop*, Cambridge, Massachusetts, USA, March 1993.

[10] R. Cole, et al, “Alphadigit Corpus”, *http://www.cse.ogi.edu/CSLU/corpora/alphadigit*, Center for Spoken Language Understanding, Oregon Graduate Institute, 1997.

[11] J. Godfrey, E. Holliman, and J. McDaniel, “SWITCHBOARD: Telephone Speech Corpus for Research and Development”, in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 517-520, San Francisco, California, USA, March 1992.

[12] J. Zhao, J. Hamaker, N. Deshmukh, A. Ganapathiraju and J. Picone, “Efficient N-gram Decoding for LVCSR”, to be submitted to the *DARPA LVCSR Workshop,* 1999.

[13] P. Clarkson and A. Robinson, “Language Model Adaptation Using Mixtures and an Exponentially Decaying Cache”, in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 799-802, Munich, Germany, April 1997.