# AN EFFICIENT TWO-PASS SEARCH ALGORITHM USING WORD TRELLIS INDEX

*Akinobu Lee* *Tatsuya Kawahara* *Shuji Doshita*

School of Informatics, Kyoto University
Sakyo-ku, Kyoto 606-8501, Japan

## ABSTRACT

We propose an efficient two-pass search algorithm for LVCSR. Instead of conventional word graph, the first preliminary pass generates "word trellis index", keeping track of all survived word hypotheses within the beam every time-frame. As it represents all found word boundaries non-deterministically, we can (1) obtain accurate sentence-dependent hypotheses on the second search, and (2) avoid expensive word-pair approximation on the first pass. The second pass performs an efficient stack decoding search, where the index is referred to as predicted word list and heuristics. Experimental results on 5,000-word Japanese dictation task show that, compared with the word-graph method, this trellis-based method runs with less than 1/10 memory cost while keeping high accuracy. Finally, by handling inter-word context dependency, we achieved the word error rate of 5.6%.

## 1. INTRODUCTION

We address an efficient search algorithm for LVCSR on multi-pass search strategy. This approach is hopeful in that, as search space is narrowed gradually by the preliminary pass, and more detailed and expensive models are applied with much less computational cost than one-pass search algorithms. Specifically, the first pass makes use of word 2-gram. Word 3-gram and inter-word context dependent model are introduced on the second pass.

Among the multi-pass algorithms, word graph methods are popular. The results of preliminary pass are symbolized as a graph form, where each arc represents likelihood and boundary time of a hypothesis word[1]. As it definitely aligns words to a certain segment of speech input, the errors due to the simple models and approximations cannot be recovered by the following rescoring process. This can be eased by introducing word-pair approximation which generates different word arcs for every preceding word. But it remarkably increases computational cost proportional to the vocabulary size.

In this paper, we propose a search algorithm using another intermediate form, called word trellis index. This form has three features. First, hypotheses can be re-aligned. It keeps track of all survived hypotheses within the beam every frame. As word boundaries are treated non-deterministically, we can get accurate N-best hypotheses on the later pass. Second, these hypotheses are indexed by frame. As they can be referred by frame (not bound to hypothesis-word), it is possible to predict next words even after boundaries are shifted by re-alignment. This feature enables us to perform an efficient stack decoding search on the later pass under large vocabulary task. Third, context dependency can be handled on the later pass. It allows simple "1-best approximation" (assume no dependency) rather than the expensive word-pair approximation.

The proposed search algorithm is mainly compared with word-pair methods and evaluated on 5,000-word Japanese dictation task (JNAS corpus).

## 2. APPROXIMATIONS IN LVCSR

### 2.1. 2-gram Factoring with Tree Lexicon

Word lexicon is normally tree-organized for reducing its size in LVCSR, but each 2-gram scores cannot be determined until the word end (leaf) nodes. So factoring of language score to each node is normally used to give linguistic constraint as early as possible. Typically, the factoring value is defined as follows:

$$\pi(s|v) := \max_{w \in W(s)} p(w|v) \qquad (1)$$

Here $s$ is a node in the lexicon tree, $W(s)$ is a set of words that share prefix at $s$, and $v$ is the last word hypothesis. As the search proceeds, factored scores are updated towards the leaf node[3].

This is an optimal method in that the factoring score is guaranteed to be larger than the actual 2-gram score, provided that $v$ remains the same through the current word.

### 2.2. N-best Approximation

Matching length of words varies depending on the context by co-articulation effect. This dependency may affect the entire sentence (sentence-dependent), but as it is difficult to deal with separate hypotheses for all possible contexts in LVCSR, approximations that limits the range are used. Here, two methods are compared.

### Word-pair approximation

Assume dependency on only a preceding word[4]. In a frame-synchronous search, word lexicons are multiplied in parallel corresponding to the preceding word, and each leafs and roots are connected to build a recognition network, either dynamically or statically.

Coupled with word 2-gram, equation (1) is clearly satisfied at each copied lexicon, so factoring works well. But making copies of whole lexicon for each word costs much memory size especially in large vocabulary.

### 1-best approximation

Assume no dependency. Only one lexicon is used, but the scores contain boundary error for word hypotheses other than the best sequence. So the acoustic score is not accurate for all hypotheses.

Moreover, together with the tree lexicon, the factoring error is caused. Because hypotheses that have different preceding words are merged in a single tree, the best one $\hat{v}$ overrides others within $W(s)$. So the substituted value $\pi(s|\hat{v})$ spoils optimality of factoring and causes language score error.

## 3. WORD TRELLIS INDEX

Based on the viewpoint in the previous section, we first review the conventional word graph method. Then, we propose word trellis index method.

### 3.1. Word Graph

Word graph is a compact representation of N-best candidates. The arcs represent word hypotheses[1]. As it represents each word hypothesis bound to a certain segment of speech input, re-alignment on the later pass is essentially not allowed. In other words, it lacks information about the non-determinacy of word boundaries. Even if time constraint can be ignored on the later pass, their possible sequences are still bound by results of the first pass.

The context dependency of word boundaries must be handled on the first pass and thus word-pair approximations are ordinary adopted. By the same reason, more precise and expensive acoustic models should be also applied on the first pass. These causes much computational cost as vocabulary gets large.

### 3.2. Word Trellis Index

We propose applying a trellis form in LVCSR. Trellis is a kind of intermediates in which all paths of Viterbi scores at every frame are kept[2]. Actually only the word-end nodes that survived within the beam are sufficient. At the later search, it serves as a heuristics.

As it keeps all survived word-ends per frame instead of those in the N-best sequence, the time constraints are represented non-deterministically and not strictly bound to any word sequence. The word boundaries are determined on the second search using stack decoder, where the trellis is connected as backward heuristics.

There are two advantages in re-alignment on the later pass. First, the space-narrowed second pass realizes fully sentence-dependent scoring and recovers approximation error. Second, expensive word-pair approximation is not needed. The different word-ends dependent on the previous word context will hopefully be included as trellis nodes in different time-frames. So even 1-best approximation will be enough for those ends to survive. Both acoustic and linguistic errors are recovered on the second pass that performs realignment and rescoring.

However, the trellis itself is a set of nodes and scores, and does not have an explicit predictive information for the second pass. Vocabulary-level constraint alone will suffice on single word recognition[5], or grammar-level constraint will do in small vocabulary CSR[2]. In LVCSR, whose search space is huge, space-narrowing based on preliminary acoustic matching is essential.

Here, we extend the trellis form to be applicable to LVCSR. To predict next words explicitly from the preliminary results of the acoustic matching, we store the indexes of words for every frame whose end-nodes survived in beam. In the second pass, only words within the index at the end-frame of the hypothesis are expanded.

To estimate the end frame, the word beginning frame corresponding to each end node is also stored. This word expansion algorithm is shown in Figure 1. The end frame may not be correct as the matching length may vary in the second pass, but practically their differences are absorbed by the trellis property that word-ends appear successively in a certain range of frames.

The second pass performs a best-first stack decoding search in backward (right-to-left) direction, using the word trellis index as both heuristics and word prediction. The score for a hypothesis $n$ is given as:

$$f(n) = g(n) + h(n) \tag{2}$$

where $g(n)$ is a forward score and $h(n)$ is a backward score on the first pass. As search proceed per word, word-level constraints such as word 3-gram and inter-word context dependent model can be applied easily.

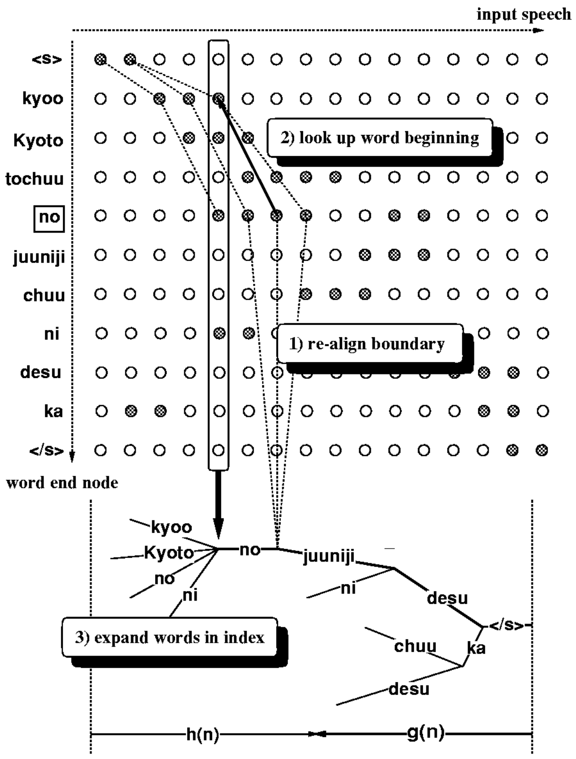


Figure 1: Word trellis index and its use in word expansion on the second pass

Table 1: Search algorithms evaluated

| first pass | intermediate form | |
|---|---|---|
| | word trellis index | word graph |
| word-pair | word-pair+trellis | word-pair+graph |
| 1-best | 1-best+trellis | |

## 4. EXPERIMENTAL RESULTS

We implemented a portable speech recognition engine named JULIUS, which can deal with both 1-best and word-pair approximation, and can use both word trellis index and word graph as an intermediate form. Table 1 lists combinations of methods evaluated. The baseline is a typical word graph method (word-pair+graph). In the comparison, inter-word context dependency is not handled for convenience.

### 4.1. Condition

The task is 5,000-word dictation of Japanese Newspaper Article Sentences (JNAS) corpus collected by Acoustical Society of Japan[6]. Language model is a word 2-gram for the first pass and word 3-gram in reverse direction for the second pass[7] trained by the Mainichi newspaper articles of 45 months. Acoustic model is a speaker-independent, gender-dependent triphone HMM[7] trained by ASJ corpus. It has 2,110 states and 16 mixtures. Lexicon contains 5,005 words with 7,451 entries considering variety of pronunciation.

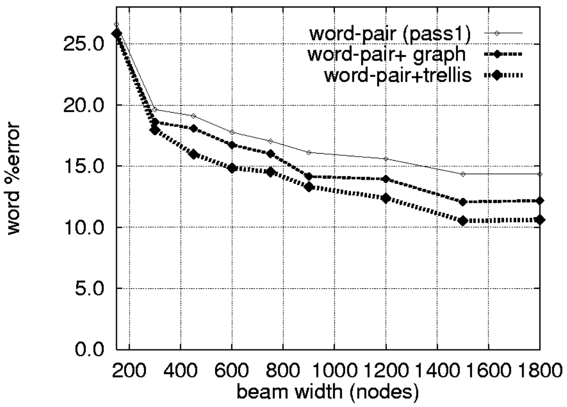


Figure 2: Word %Error: trellis vs. graph

We used test set of 100 samples by 10 speakers. The word accuracy is calculated by Katakana transcription.

### 4.2. Trellis vs. Word Graph

First, intermediate forms are compared. Both uses word-pair approximation on the first pass. Word %error per beam width is shown at Figure 2. Results on the first pass is also plotted here. The word trellis index achieves better accuracy (10.5%) than word graph (12.0%) given enough beam width. It is shown that the word-pair approximation includes some errors, and they are recoverable on the second pass by re-alignment with word trellis index.

### 4.3. Word-Pair vs. 1-Best Approximation

Next, two approximation methods are compared to examine how the errors influence to the final result. The results with word trellis index are shown in Figure 3. With 1-best approximation together with tree lexicon, both approximation and factoring errors increase recognition failure on the first pass by 5.0%. However, they are recovered to 1.7% on the second pass, that is comparable to the baseline method. Thus, it is confirmed that even with simple 1-best approximation, trellis index does not lose word boundaries which will make up the best sequence and finally gives as same accuracy as the word-pair approximation does.

In addition, when the beam width is relatively small, 1-best approximation gets better accuracy by 1%. This suggests that with word-pair approximation, same word hypotheses with different contexts occupy the beam, which are merged in word trellis index.

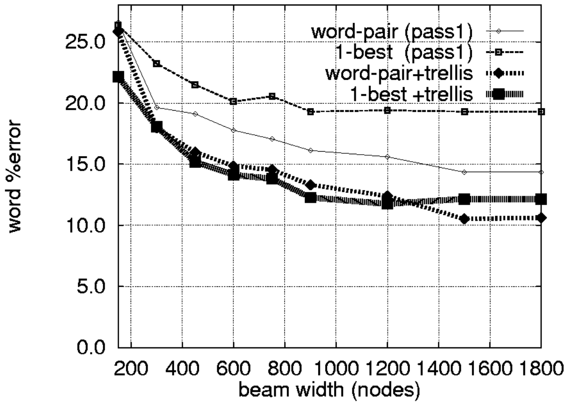
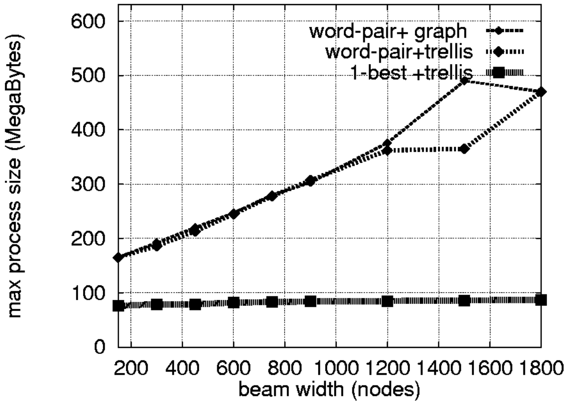


Figure 3: Word %Error: word-pair vs. 1-best



total memory size

Figure 4: Computational cost

### 4.4. Efficiency Comparison

Next, we compare the algorithms with respect to efficiency. Figure 4 shows total memory size needed. Compared to the baseline (word-pair+graph), word trellis index (word-pair+trellis) needs trellis reconnection procedure for re-alignment on the second pass, but it costs a little. And the introduction of 1-best approximation (1-best+trellis) significantly reduces computation cost. The average CPU time becomes almost 2/3, and the maximum workspace size needed for the search is reduced from over 400MB to nearly 30MB (plus 56MB for models). This difference arises from the copying of lexicon on the first pass and will grow as vocabulary becomes large.

Thus the proposed search method (word trellis index + 1-best approximation) is proved to be superior in that it performs far more efficiently while keeping high accuracy.

### 4.5. Final Result

Finally, we pursue the best performance of the decoder. Now inter-word context dependency is handled and triphone HMM is updated to have 3,000 states. As a result, the word error rate of 5.6% is achieved.

## 5. CONCLUSION

A two-pass search algorithm with trellis interface is presented. The trellis form is extended to word trellis index that has the frame-indexed active word list and corresponding time-frame information to be applicable for LVCSR.

Compared to the conventional word graph method, word trellis index can recover the errors caused by the approximations. Thus, simple 1-best approximation instead of word-pair approximation is sufficient to achieve almost the same accuracy while computational cost is remarkably reduced. Word accuracy reaches 94.4% in the best case.

**Acknowledgment:** the Japanese phone model and language model are provided in Japanese Dictation Toolkit[7].

## References

[1] H. Ney and X. Aubert : A Word Graph Algorithm for Large Vocabulary Continuous Speech Recognition, *Proc. ICSLP*, pp1355–1358 (1994).

[2] F.K.Soong and Eng-Fong Huang : A Tree-Trellis Based Fast Search for Finding the N-Best Sentence Hypotheses in Continuous Speech Recognition, *Proc. IEEE-ICASSP*, pp705-708 (1991).

[3] J.J.Odel, V.Valtchev, P.C.Woodland and S.J.Young : A One Pass Decoder Design for Large Vocabulary Recognition, *Proc. ARPA Human Language Technology Workshop*, pp.405–410 (1994).

[4] R.Schwartz et al. : A Comparison of Several Approximate Algorithms for Finding Multiple (N-best) Sentence Hypotheses, *Proc. IEEE-ICASSP*, pp701–704 (1991).

[5] J.K.Chen, F.K.Soong and L.S.Lee : Large Vocabulary Word Recognition Based on Tree-Trellis Search, *Proc. IEEE-ICASSP*, pp137–140 (1994).

[6] K.Itou, M.Yamamoto, K.Takeda, T.Takezawa, T.Matsuoka, T.Kobayashi, K.Shikano and S.Itahashi : The Design of the Newspaper-Based Japanese Large Vocabulary Continuous Speech Recognition Corpus, *Proc.ICSLP* (1998).

[7] T.Kawahara, T.Kobayashi, K.Takeda, N.Minematsu, K.Itou, M.Yamamoto, A.Tamada, T.Utsuro and K.Shikano : Sharable Software Repository for Japanese Large Vocabulary Continuous Speech Recognition, *Proc.ICSLP* (1998)