J. Computer Science & Technology, 15(5): 461-471, Sept. 2000

IMPROVING THE SYLLABLE-SYNCHRONOUS NETWORK SEARCH ALGORITHM FOR WORD DECODING IN CONTINUOUS CHINESE SPEECH RECOGNITION

ZHENG Fang(郑方), *WU Jian*(武健) *and SONG Zhanjiang*(宋战江)

Center of Speech Technology, State Key Laboratory of Intelligent Technology and Systems,

Department of Computer Science and Technology, Tsinghua University, Beijing, 100084, P.R. China
fzheng@sp.cs.tsinghua.edu.cn

ABSTRACT

The previously proposed syllable-synchronous network search (SSNS) algorithm plays a very important role in the word decoding of the continuous Chinese speech recognition and achieves a satisfying performance. Several related key factors that may affect the overall word decoding effect are carefully studied in this paper, including the perfecting of the vocabulary, the big-discount Turing re-estimating of the N-Gram probabilities, and the managing of the searching path buffers. Based on these discussions, corresponding approaches to improve the SSNS algorithm are proposed. Compared to the previous version of SSNS algorithm, the new version decreases the Chinese character error rate (CCER) in the word decoding by 42.1% across a database consisting of a large number of testing sentences (syllable strings).

KEYWORDS: large-vocabulary continuous Chinese speech recognition, word decoding, syllable-synchronous network search, word segmentation

1. INTRODUCTION

A large-vocabulary continuous speech recognition (LVCSR) system often consists of two primary parts: the acoustic model (AM) and the language model (LM). By the Bayesian rule, the aim of recognition is to find the most likelihood word string W^* that satisfies

$$W^* = \arg\max_W P(W|A) = \arg\max_W P(A|W)P(W) \qquad (1)$$

where A is acoustic signal and W, one possible word string, is one output candidate of the AM stage as well as the input of the LM stage. $P(A|W)$ is the conditional probability of the utterance A given the word string W calculated by AM while $P(W)$ is the word string probability calculated by LM.

For LVCSR in Chinese, it is different. Chinese is a syllabic language. A Chinese sentence is a string of Chinese words. Each word consists of one or several Chinese characters, and mostly each character in a word with definite meaning corresponds to a unique Chinese syllable in pronunciation.

The following three major reasons go against the use of Equation (1), i.e., taking words as the common units (output units of AM and input units of LM) between AM and LM as in western languages.

Firstly, the homonym and homograph phenomena are common. There are just about 400 toneless or 1,300 toned *syllables* but more than 6,700 frequently used *characters* in Chinese. Therefore each syllable (the pronunciation) will be shared by several characters (the pictograph). On the other hand, each character may correspond to several different syllables depending on different word contexts.

Secondly, although words are basic semantic units the word boundaries are hard to determine in a given sentence. A multiple-character word can be regarded as a concatenation of shorter words recursively. Different boundary assumptions sometimes just increases the word segmentation complexity when no semantic conflict arises, but sometimes causes semantic conflict due to the uncertainty of the word boundaries. This is a problem of Chinese word segmentation, or so-called "word decoding".

Thirdly, for real-world applications there are many kinds of accents in the Chinese language even for standard Chinese. For each accent, there possibly exists a set of syllable mappings between this accent and the standard Chinese. For example, a Hong Kong person often pronounces "zhi" as "ji".

Because of these factors, choosing words as the common units will result in much redundancy in acoustic searching paths. Instead, Chinese syllables are taken as the common units and an efficient algorithm, named syllable-synchronous network search (SSNS) algorithm, was proposed in our previous paper [1].

We will first overview the proposed SSNS algorithm in Section 2, propose some methods to improve it in Section 3, give the experimental results in Section 4 and come to the conclusion in Section 5.

2. SSNS ALGORITHM

The SSNS algorithm is based on such a two-stage continuous Chinese speech recognition structure described by the following equation

$$W^* = \underset{W}{argmax}\ P(A \mid W)P(W) = \underset{W}{argmax}\ P(A \mid S)P(W)\Big|_{S=C(W)} = \underset{W:S=C(W)}{argmax}\ P(A \mid S)P(W) \quad (2)$$

where S stands for a syllable string as a result from AM stage, and $C(W)$ for the concatenation of all syllable strings corresponding to every Chinese words in the word string W. This idea aims at taking the Chinese syllables instead of words as the common units between AM stage and LM stage.

In this Chinese LVCSR structure, the output of the AM stage is the Chinese syllable lattice (CSL). A syllable based word search tree (WST) is used to describe the syllable hierarchy of the vocabulary (of Chinese words) while the tri-gram probabilities are used to reflect the relationship of every successive three Chinese words. The SSNS algorithm, via the accumulated tri-gram probabilities for Chinese words, provides a maximum likelihood match between the CSL and the loop WST.

2.1 CHINESE SYLLABLE LATTICE (CSL)

The CSL is a kind of representation of the acoustic output candidates based on the Chinese syllables. The syllables are the common units connecting the acoustic processing stage and the language processing stage.

As previously stated like in Equation (1), our LVCSR uses a two-stage Chinese syllable based structure. The CSL, see Figure 1 as an example, is obtained by the Viterbi search [2] or frame synchronous network [3] based acoustic decoding procedure among a Syllable Search Tree (SST), whose nodes can be either 22 Chinese initial plus 38 Chinese finals, or 40 Chinese phonemes [4].

The CSL should contain the correct path of the uttered sentence but it should be as small as possible so as to decrease the load in LM stage. Though it is a more important part, we won't focus on it because it is nothing to do with the SSNS algorithm itself.

2.2 VOCABULARY AND WORD SEARCH TREE

Because the vocabulary contains the morphology information of the Chinese language, how the vocabulary words are chosen and how large the vocabulary is are very important.

In our system, the vocabulary consists of two parts, the first part is called the system vocabulary (SVOC) and the second is the user vocabulary (UVOC). The design of the SVOC follows the following rules:

(1) the words should not be too long (no longer than 4 Chinese characters), of course the SVOC should not consist of monosyllable words only;
(2) the compound words should be excluded as possible as you can because these words can be segmented into a sequence of sub-words that are also the SVOC words (but of course these sub-words should not be mainly monosyllable words);
(3) not all 6,700 Chinese characters (mostly monosyllable words) are included, only the most frequently used monosyllable words are included in the system word set; and,
(4) punctuation marks are treated as special SVOC words.

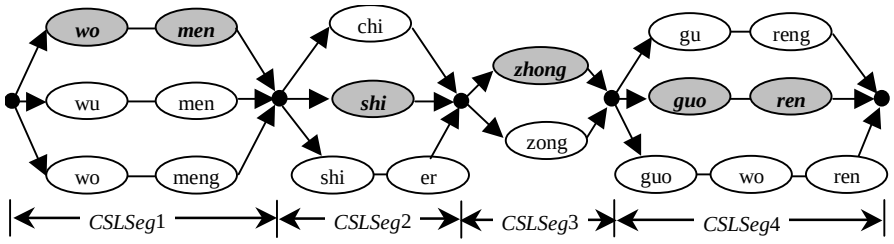

FIGURE 1. AN EXAMPLE FOR THE CHINESE SYLLABLE LATTICE (CSL).

(The original spoken sentence is "wo3 men2 shi4 zhong1 guo2 ren2 (we are Chinese)".)

Based on the above principles, our system word set is designed to include 50,624 Chinese words, where there are 6,201 monosyllable words (12.3%), 37,976 bi-syllable words (75.0%), 1,615 tri-syllable words (3.2%) and 4,832 quad-syllable words (9.5%). This vocabulary is over two times bigger than our previous one [5][1].

The UVOC words can be as long as 10 Chinese characters, and currently the number is limited only by the memory.

In order to improve the word decoding efficiency, the vocabulary is organized into a syllable-based word search tree (WST) [1][6]. It is designed to reflect the relations among all the in-vocabulary words so that the redundancy for both the vocabulary storage and the searching consumption are reduced. In this tree, all nodes except the virtual root node and the leaf nodes are called *Syllable Nodes*, because they each contain the information of a syllable of a word. This tree is established recursively by this rule: all words whose first n syllables are exactly the same will

share a unique *n*-th level *Syllable Node* and the syllable stored in this node is exactly the *n*-th syllable of these words. The child node of the *n*-th level node (either the root node or any *Syllable Node*) is one possible successive syllable of the current node to form the first (*n*+1) syllables of a word, it can be either an *Syllable Node* (word length exceeding *n* syllables) or a *Leaf Node* (word length exactly *n* syllables). A *Leaf Node* does not contain syllable information but the information of the word whose corresponding syllable string is exactly the same as the string of sequential syllables contained in the corresponding *Syllable Nodes* covered by the route travelling from the root node to its parent *Syllable Node*. Because a *Leaf Node* has no child node, reaching a *Leaf Node* causes an accumulation of word N-Gram probabilities and an extended search from the *Root Node* during the word decoding. Figure 2 is an example of WST, where there are only 7 words.

By travelling through the WST, any word segmentation, i.e. word-decoding, for a given word string can be easily covered, for example, " 中国 " can be covered by travelling along Route "*Root Node*→ 'zhong' → 'guo' → Φ(中国)" (as a single word) or along Route "*Root Node* → 'zhong' → Φ (中) → *Root Node* → 'guo' → Φ(国)" (as a concatenation of two words). As illustrated in this example, the WST is looped during word decoding procedure.

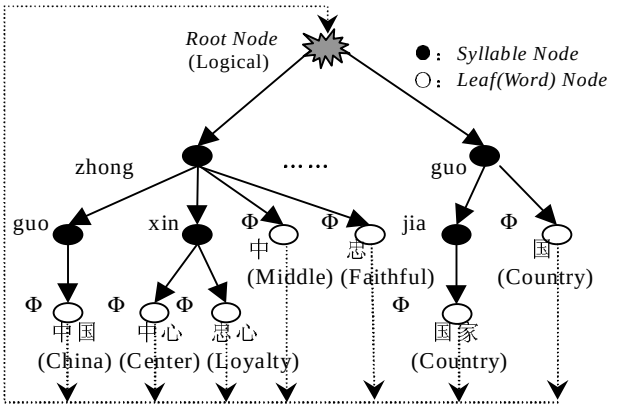

FIGURE 2. A PARTIAL WORD SEARCH TREE (WST).

An actual WST is more complicated, for the vocabulary is very large and contains many relatively long Chinese words. In such a tree the travelling direction is always from the parent node to its child node(s), so it can be stored in a linear data structure, i.e., an array of nodes. By using a well-designed algorithm, it takes only several seconds to establish a WST for 50,000 words, and takes almost no time to incrementally establish such a tree when users insert or delete user words.

2.3 N-GRAM BASED LANGUAGE MODELING

Given a definite Chinese syllable string $S = \{s_1, \cdots, s_L\}$, there would be many Chinese word string candidates due to the uncertainty of the Chinese word boundaries. Among these word string candidates $\{W | C(W) = S\}$, there should be only one correct word string. We take the word string with most likelihood score (MLS) as the final word string corresponding to the given syllable string. (The word decoding among the CSL will be more complicated.) Assume a word string is $W = \{w_1, \cdots, w_N\} \overset{def}{=} w_1^N$, its MLS is defined as the probability of the word string, which is simplified as

$$P(W) \approx P(w_1)P(w_2|w_1)\prod_{n=3}^{N} P(w_n|w_{n-2}, w_{n-1}) \tag{3}$$

This is the well-known tri-gram model. In the Chinese language, there are about 50K commonly used words. For such a large vocabulary, the probability matrix is quite sparse due to the lack of training data. Some tri-grams which might make sense but do not occur in the training data are regarded as impossible, a case which may degrade the recognition rate greatly. In order to give the unseen word combinations reasonable probability estimation, we employ a Turing's method [7] and modify it to be more practical one [8]. The modification of Turning's method greatly reduces the perplexity of language model.

2.4 SYLLABLE-SYNCHRONOUS NETWORK SEARCH (SSNS)

In this section, we will briefly describe the SSNS algorithm. For convenience, we will define a data structure named as the search path to remember the instantaneous matching information between the already-processed partial syllable string and its corresponding partial word string. Mainly, each search path contains the following fields:

- *CSLNode* – pointer to current node in the CSL;
- *WSTNode* – pointer to current node in the WST;
- *PartWordString* – partial word string already decoded; and
- *LLScore*: the accumulated log likelihood N-gram score of the partial word string.

The SSNS algorithm has the following steps (there will be some practical modifications in implementation).

STEP 1. INITIALIZATION.

Creating one search path, with *WSTNode* pointing to the root node of WST, *CSLNode* pointing to the very beginning of the CSL, and *PartWordString* and *LLScore* reset.

STEP 2. FORWARDING ONE SYLLABLE.

(1) Checking each search path: if the *CSLNode* is still inside the same *CSLSeg* (as illustrated in Figure 1), it has one unique *CSL* node successor; if the *CSLNode* just passes across the end of the path in the *CSLSeg*, it takes all the starting nodes in each parallel *CSL* paths of next *CSLSeg* as its *CSL* node successors.

(2) Forwarding *CSLNode* to each of its successors, and one new search path is generated for each successor.

Meanwhile, old search paths are removed. (This is the syllable-synchronous forwarding operation in CSL.)

(3) Forwarding the *WSTNode* according to the current syllable contained in each new-generated search path's *CSLNode* (just updated in Sub-Step (2)): (a) if *WSTNode* is not a leaf node and we can find the current syllable among its child nodes, forward it to this node, otherwise remove this path; (b) if *WSTNode* is a leaf node, a new word is generated, so we first append this word into the *PartWordString* and update *LLScore* according to the N-gram calculation, and then warp the *WSTNode* back to the root so as to perform the (3.a) operation. (This is the syllable-synchronous forwarding operation in WST.)

STEP 3. PRUNING PATH.

Pruning all those search paths that are less competitive (with lower *LLScore*, can be predicted to be pruned in the next forwarding operation, and so on.)

STEP 4. CHECKING FOR END OF CSL.

If all paths stop at the end of the CSL go to Step 5, otherwise go back to Step 2 for next search.

STEP 5. FINALIZATION OF THE SEARCH.

The search path whose *WSTNode* ends at one WST leaf node and who has higher *LLScore* value is one of the final candidates. The *PartWordString* contains the resulted word string.

The SSNS algorithm together with WST and N-gram based language model can easily solve the homonym and word segmentation problem in Chinese syllable-to-word conversion of the Chinese LVCSR as well as in the character-to-word conversion of the Chinese optical character recognition. Its idea is also useful in the word segmentation for Chinese sentences.

2.5 SOLUTION TO ACCENT ROBUSTNESS

In Chinese, there are many kinds of regional accents all over China and overseas even the speakers themselves tend to speak in standard Chinese (Mandarin). Figure 3 shows two examples of accents, where *A* and *B* stand for two different syllables. Case I is something like the different speaker issue, which is easy to be solved via an embedded multiple-model (EMM) scheme [9] or context-dependent modeling techniques in the acoustic processing layer. But Case II is possibly Chinese accent specific. In a certain accent for Case II, some syllables are mapped into quite different syllables (not just similar as in Case I), for example, a southern Chinese speaker may pronounce syllable 'zhi' completely into 'zi'. So we propose different approach to the second case.

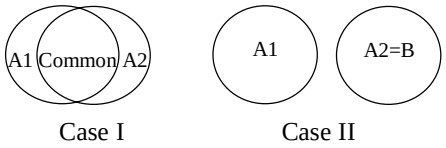

FIGURE 3. TWO KINDS OF ACCENT EXAMPLES IN CHINESE.

The Case II accent problem can not be solved in the acoustic layer. Our proposed solution is to apply the concept of the Chinese fuzzy syllable set to the SSNS algorithm in the language-processing layer.

For clarification, we define *FSS*(*X*), the Fuzzy Syllable Set of a syllable *X*, as a set of syllables that may be pronounced into *X* in a specific regional accent. By using the knowledge of regional accents, we can list a group of accent-syllable/initial/final to Mandarin-syllable/initial/final mapping pairs as shown in Table 1 for the user to check/uncheck these options one by one. Once the user checks any pairs of fuzzy syllables/initials/finials, the fuzzy syllable set generation processor will generate all the possible fuzzy syllable pairs. E.g., if "zhi→ji" is checked then *FSS*("ji") = {"zhi", "ji"}; if "z↔zh" is checked then *FSS*("zhe")={"zhe", "ze"}, *FSS*("za") = {"zha", "za"} and so on. The Case II accent problem can be solved by the arc-splitting technique in the SSNS algorithm.

TABLE 1. EXAMPLES OF FUZZY SYLLABLE/INITIAL/FINAL MAPPING

Whole Syllable		Initial		Final	
ZHI →	JI	Z ↔	ZH	AN ↔	ANG
CHI →	QI	C ↔	CH	EN ↔	ENG
SHI →	XI	S ↔	SH	IN ↔	ING
WANG ↔	HUANG	F ↔	H		
WEN ↔	WENG	N ↔	L		
GUO →	GUI	W ↔	HU		
		Y ←	R		

3. IMPROVING SSNS ALGORITHM

It is obvious that the CSL, the WST and the estimated accuracy of the word tri-grams are three key factors in the SSNS algorithm that may affect the final performance of a LVCSR system. The searching strategy of the SSNS itself is also another factor. In this paper, we propose some approaches to improve the SSNS algorithm based on the above discussion.

In this section, the key factors will be discussed in details. The improvement methods are also given upon these discussions.

3.1 PERFECTING OF VOCABULARY

3.1.1 Size of Vocabulary

It is obvious that the size of the vocabulary is one of the key factors that may affect the performance of the Chinese LVCSR.

If the size is small the word decoding procedure will be fast and the accuracy will be high but the spoken sequence words is limited by the small vocabulary, and vice versa. The vocabulary can be neither too large nor too small. A best solution is to maintain a medium-size vocabulary consisting of most common used words with a flexible mechanism so that the users can add words freely and easily according to different application domains. The SVOC's Design Rule 2 helps to do this. That is why the words in our vocabulary are mostly bi-syllable words.

3.1.2 Monosyllable Words

According to the SVOC's Design Rule 3, we don't need to include all monosyllable words into our vocabulary. This perhaps gives rise to a mismatch between the CSL and the looped WST when there is a (true) recognized syllable that can neither match any monosyllable words nor match the syllable context in any multi-syllable words, resulting in the false path pruning in the word decoding algorithm. This case may be met frequently when the person and/or place names are spoken in the sentences. So the deletion of monosyllable words from the vocabulary should not only be based on the occurrence frequency unconditionally.

Our solution to this problem is the concept of the *Filler Syllable*.

We follow the below steps to do the special processing over the monosyllables words.

(1) List all the 418 Chinese syllables;
(2) For each syllable find a corresponding most frequently used monosyllable words;
(3) All the 418 monosyllable words can be divided into four types. Type (0) *Normal monosyllable words*: neither the Chinese characters (Hanzis) nor the syllables of these words have their particularity. There are 400 normal monosyllable words. Type (1) *Filler monosyllable words*: these syllables can be acoustically modeled but the corresponding characters can not be found in the GB2312 character set. There are 10 such words. Type (2) *Acoustically unseen monosyllable words*: these syllables are difficult or even impossible to be modeled acoustically, and they will never appear in the syllable lattice in the acoustic layer. There is no need for all the three such words to present in the vocabulary. Type (3) *Spoken monosyllable word*s: these syllables have more reasons to present in the vocabulary because they are frequently used in the spoken language. They can be regarded as special syllables. These syllables include 'dei3' ('得', to need, to have to, must), 'lia3' ('俩', two/both, not many), 'shei2' (an alternative pronunciation for '谁 shui2', who), 'tei1' ('忒'), and 'zhei4' (an alternative pronunciation for '这 zhe4', this).

Except the normal monosyllable words, most of them have no homophonic monosyllable words.

Type (2) words are unseen in the vocabulary. Type (0) and Type (3) words are included in the vocabulary. Type (1) words, whose corresponding syllables are called filler syllables, are included in the vocabulary without displayable Chinese characters. They are used to prevent the false pruning of the paths in the SSNS word decoding procedure, and they will be deleted from the final sentences.

3.1.3 Polyphonic Words

The polyphonic-word phenomenon is an issue that must be faced because there are many polyphonic words (especially monosyllable words) in Chinese language.

In our N-Gram estimation, we can see only the words' Chinese character strings (or word texts, word forms) instead of the words' syllable strings (pronunciations). No matter how many pronunciations a word has the word (or word gram) occurrence times are difficult to be distributed to these different pronunciations of the word. In this situation, the statistics is based on the word form (i.e. word text).

There is an almost irrealizable solution, which is to label all words in the training materials with their corresponding syllable strings (pronunciations) according to the sentence context so that all polyphonic words can be distinguished by their pronunciation. But the labeling would be extremely time-consuming.

It is difficult to find a realizable and satisfying solution to this problem but it is a must to solve it. As a matter of fact, there is no need to do the labeling. The aim here is only to offer a possible pronunciation-to-text mapping so that the paths including these words will not be pruned when their alternative pronunciations are present. A sub-optimal method follows these steps. (1) When counting the occurrence of words (word grams) to estimate the N-Gram probabilities, treating the words that share the same Chinese character strings (word text) as one same word with a unique word ID by which the N-Gram probabilities are accessed. (2) When building the WST, treating words that share the same word text but are different in pronunciation as different words and adding them all to this WST. (3) Assigning the same word ID to all these polyphonic words with the same word text so that they share the same N-Gram probabilities.

As a matter of fact, in this method, all the occurrence counts of a word with different pronunciations are summed up and assigned to the representative word. So it is not perfect but practical.

3.2 RE-ESTIMATION OF N-GRAM PROBABILITIES

Which paths should be kept and which should be pruned when the CSL is being matched with the loop WST are determined by the Accumulated Log Likelihood N-Gram Probability (referred to as *ALLP score* from this point forward) of each path. The bigger *ALLP* score a path has, the more possible it will be kept. *ALLP* score is accumulated at word boundaries, that is to say, when the CSL is just arriving at any leaf in the WST.

Let us denote the word sequence $w_m,\cdots,w_n$ as w_m^n. Assume the word sequence already decoded and stored in a path is w_1^k, then the *ALLP* score of this path is

$$\ln P\left(w_1^k\right) \approx \ln Uni\left(w_1\right) + \ln Bi\left(w_1^2\right) + \sum_{n=3}^{k} \ln Tri\left(w_{n-2}^n\right) \quad (4)$$

where $Tri\left(w_{n-2}^n\right) = P\left(w_n | w_{n-2}, w_{n-1}\right)$, $Bi\left(w_1^2\right) = P\left(w_2 | w_1\right)$, and $Uni\left(w_1\right) = P\left(w_1\right)$. This is the well-known tri-gram model. In the Chinese language, there are about 50~60K commonly used words. For such a large vocabulary, the probability estimate matrix is quite sparse due to the sparseness of the training data. In order to give every unseen word sequence a reasonable probability estimate, we employ a Turing's method [7] and modify it to be a more practical one [8]. The modification of Turing's method greatly reduces the perplexity of language model. The language model based on this method is the baseline in this paper.

For a vocabulary of 50~60K words, the tri-gram computation (either directly accessed or re-estimated by means of Turing's method) takes a very long time and makes the LVCSR not in real-time. So a more practical and efficient method is proposed in this paper. In consideration of the sparseness of the tri-gram probability matrix, if the number of times a tri-gram w_1^3 occurred in the training text $c(w_1^3)$ is not greater than a constant $k = 5$, we will re-estimate its occurrence probability as a "discounting" of that of the corresponding lower level bi-gram w_2^3. The discount coefficient $d_r'(w_1^2)$ related to this tri-gram is pre-calculated according to the distribution of all the counts that are less than or equal to k. The idea can be described by Equations (5) and (6).

$$Tri'(w_1^3) = d_r'(w_1^2) \cdot Bi(w_2^3), \quad if\ c(w_1^3) \le 5 \quad (5)$$

$$d_r'(w_1^2) = \frac{\sum_{w_1^3 : c(w_1^3) \le 5} c(w_1^3)}{\sum_{w_1^3} c(w_1^3)} \cdot \frac{1}{\sum_{w_2^3 : c(w_2, w_3) = 0} Bi(w_2^3)} \quad (6)$$

It is referred to as a *Big-Discount* re-estimation because those grams with occurrence counts not greater than a predefined big constant $k = 5$ are regarded as unseen ones. It not only makes the word decoding procedure much faster, but also improves the decoding accuracy. The details of this Big-Discount Turing's method will be present in other paper.

3.3 SEARCHING ALGORITHM

We refer to the word decoding procedure as a search in the CSL. The SSNS algorithm makes the CSL travel in the loop WST circularly and at last gives the word decoding sequence with the maximum *ALLP* score as the final decoded sentence[1]. It is very similar to the frame synchronous network search (FSNS) algorithm used for the state decoding in the acoustic stage[3]. In this section, several improvements will be made to the searching algorithm.

3.3.1 Number of Paths

Paths are used to remember the current matching status between the CSL and the loop WST when forwarding the CSL among the WST syllable-by-syllable. If the number of paths is large enough, the SSNS algorithm may almost be able to perform the full search and the word decoding result will be near to being optimal but time-consuming, otherwise the result will be less time-consuming but (a little bit) far from being optimal.

The number should be suitable. A compromise should be made to balance the efficiency and the result.

3.3.2 Managing the Path Buffers

Unlike in the FSNS procedure where the acoustic matching probabilities are accumulated at each frame, in the SSNS procedure the *ALLP* score (of course of words) are not always accumulated at each syllable but only at the leaf nodes (or in the other words, at word boundaries). The situation is quite different from that in the FSNS algorithm. This results in the unfair probability comparisons among paths.

Here is an example to show such an inequality.

Given a syllable string as a degenerated CSL to be decoded into word sequence, say "*wai dian ping shu cheng zhong guo ren min zu yi shi qiang* (外电评述称中国人民族意识强)", two partial paths are as follows (the item on the right of '⇒' stands for the next syllables/words to be matched):

(1) '*wai*' → '*dian*' → Φ (外电) → '*ping*' → '*shu*' → Φ (评述) → '*cheng*' → Φ (称) → '*zhong*' → '*guo*' → '*ren*' → '*min*' ⇒ Φ(中国人民) + '*zu*' …

(2) '*wai*' → '*dian*' → Φ (外电) → '*ping*' → '*shu*' → Φ (评述) → '*cheng*' → Φ (称)→ '*zhong*' → '*guo*' → 'ren' → Φ (中国人) → '*min*' ⇒ '*zu*' …

The *ALLP* scores for Paths 1 and 2 are

$$ALLP(1) = LnTri(A, B, C), \text{ and}$$

$$ALLP(2) = LnTri(A, B, C) + LnTri(B, C, E)$$

respectively, where *A*='外电', *B*='评述', *C*='称', *D*='中国人民', *E*= '中国人'. It is obvious that $ALLP(1) > ALLP(2)$. But Path 1 may lead to a wrong result because paths with lower *ALLP* scores take the risk to be pruned and the probability accumulation takes place at word boundaries gives higher priority to the long words in the path sorting.

A Dual-Path-Buffer scheme offers a good solution. In this scheme, one path buffer is used to store those sorted paths that end at word boundaries and another is used for those sorted paths that is still wandering inside words. The first buffer is called the *Complete-Word Path Buffer* (CWPB) and the latter the *Partial-Word Path Buffer* (PWPB). Paths in two buffers are compared and pruned individually.

4. EXPERIMENTAL RESULTS

4.1 EXPERIMENTAL CONDITION

In our experiments, the size of system vocabulary is 50,624 Chinese words of length ranging from 1 to 4 Chinese characters (syllables). The N-Gram statistics is also based on such a large vocabulary.

In order to evaluate the performance of the word decoding only, the testing data are sentences with their corresponding unique syllable strings labeled. That is to say, each time the input CSL is degenerated into an exact syllable string.

There are 1,559 sentences consisting of totally 22,083 Chinese syllables (characters). These syllable strings form the testing bed.

The accuracy is the percentage of how many correct (i.e. matched) words out of the original words are outputted.

4.2 EXPERIMENTAL RESULTS

In this section, we give the results of our step-by-step incremental experiments.

4.2.1 Baseline & Big-Discount Re-estimation

The baseline experiment is conditioned on (1) the original SSNS algorithm and (2) the standard Turing's tri-gram re-estimation [10][8]. Table 2 lists the experimental result of the baseline experiment as well as the result of the experiment where the Big-Discount Turing Re-estimation is adopted where the size of the path buffer is 20 for both experiments. The experiments are done under the Pentium II 450MHz PCs.

TABLE 2. BASELINE AND BIG-DISCOUNT RE-ESTIMATION EXPERIMENTS

Experiment	Accuracy	Processing Speed (syllables per second)
Baseline	93.49%	12
Big-Discount	93.90%	75

4.2.2 Dual-Path-Buffer Scheme

This experiment is based on the Big-Discount Turing's Re-estimation and designed to test the effect of the use of the dual path buffer scheme. We distribute the original 20 paths into two buffers, one is for the complete-word path buffer (CWPB) and the other is for the partial-word path buffer (PWPB). The total size of the two buffers is the same as that in the original experiments, i.e 20. Table 3 gives the results. Obviously any of the three kinds of path distributions can achieve a better word decoding performance, but the path distribution of equal buffer sizes reaches the highest accuracy.

In this experiment, the total buffer size remains the same, therefore any kind of Dual-Path-Buffer scheme does not result in extra time complexity.

TABLE 3. DUAL-PATH-BUFFER EXPERIMENT

Size of Buffer (# of paths in buffer)		Accuracy
CWPB	PWPB	
10	**10**	**95.41%**
15	5	95.38%
5	15	95.16%

4.2.3 Perfecting the Vocabulary

According to the discussion in Section 2.2, we consider the filler monosyllable words so that the least frequently used syllables/characters (often seen in person/place names) can be matched without false pruning of paths during the SSNS procedure. We also add polyphonic words to the vocabulary so that different pronunciations of words can be considered. These meticulous steps improve the overall word decoding accuracy, see Table 4 for the result.

Obviously, although we add filler monosyllable words and polyphonic words to the vocabulary, the WST data structure ensure that the size of WST is not enlarged too much, and the added parts can be ignored if considered in percentage. This makes no more time consumption in SSNS search.

TABLE 4. VOCABULARY PERFECTING EXPERIMENT

Perfecting the Vocabulary:	Accuracy
Using filler monosyllable words and polyphonic words	96.23%

5. CONCLUSION

In closing, we come to the following conclusions:

(1) At least one corresponding monosyllable word of each Chinese syllable should be kept in the vocabulary, if the vocabulary can not include all possible words, so as to ensure those paths that may stop by such syllables will not be falsely pruned. If there is no such Chinese character (namely unseen in computer), the filler syllables should be used to filter these least frequently used monosyllable words and to ensure such paths to forward smoothly.

(2) Before polyphonic words can be considered in the statistics of the N-Gram probabilities, these words should be added to the word search tree and share the same gram probabilities.

(3) The Big-Discount Turing's Method can be used to speed up the word decoding procedure and to improve the accuracy. In this method proposed in this paper, the tri-grams with observed occurrence in the sample text less than a given big constant are regarded as "unseen", and the tri-gram probability estimates are discounted from the corresponding lower level bi-gram probabilities.

(4) During the syllable synchronous network search procedure, those paths that are just travelling by the word boundaries and those paths that is travelling inside the words should be

compared and pruned individually in two buffers (referred to as Dual-Path-Buffer in this paper). This can make the path pruning a little bit far from being unfair where the long words are given higher priority to be kept. Because the total buffer size remains the same, this modification will not cause extra loads in the search.

The above means are proved helpful to improve the overall SSNS performance, and the experimental results show that the Chinese character error rate (CCER) in the word decoding can be decreased by 42.1%.

6. REFERENCES

[1] Zheng, F., "A syllable-synchronous network search algorithm for word decoding in Chinese speech recognition," *IEEE International Conf. on Acoust., Speech and Signal Processing (ICASSP)*, pp. II-601~604, March 15~19, 1999, Phoenix, USA

[2] Viterbi A.J. "Error bounds for convolutional codes and an asymptotically optimum decoding algorithm," *IEEE Trans. on IT-13(2)*, Apr., 1967

[3] Lee C.-H. and Rabiner L. R. "A frame-synchronous network search algorithm for connected word recognition," *IEEE Trans. on ASSP*, 37(11): 1649-1658, Nov. 1989

[4] Zheng F., "Studies on approaches to keyword spotting in unconstrained continuous speech." *Ph.D. dissertation*: Computer Sci. & Tech., Tsinghua University, May 1997.

[5] Zheng F., Mou X.-L., Xu M.-X., *et al*. "The implementation of a speech-to-text editor," *5th National Conference on Man-Machine Speech Communication (NCMMSC-98)*, 280-285, 1998 (In Chinese)

[6] Zheng, F., Song, Z.-J., Xu, M.-X., Wu, J. *et al*, "EasyTalk: a large-vocabulary speaker-independent Chinese dictation machine," *EuroSpeech'99*, Vol. 2, pp.819-822, Sept. 1999, Budapest

[7] Nadas A. "On Turing's formula for word probabilities," *IEEE Trans. on ASSP*, Vol. ASSP-33, No. 6, June 1985

[8] Mou X.-L., Zhan J.-M., Zheng F. and Wu W.-H. "The back-off algorithm based N-gram language model," *5th National Conference on Man-Machine Speech Communication (NCMMSC-98)*, 206-209, 1998 (In Chinese)

[9] Zheng, F., Mou, X.-L., Wu, W.-H., and Fang D.-T. (1998), On the embedded multiple-model scoring scheme for speech recognition. *International Symposium on Chinese Spoken Language Processing (ISCSLP'98)*, ASR-A3, pp.49-53, Dec.7-9, 1998, Singapore

[10] Katz, S. M., "Estimation of probabilities from sparse data for the language model component of a speech recognizer," *IEEE Trans. on ASSP*, 35(3): 400-401, March 1987