THE BELL SYSTEM TECHNICAL JOURNAL
Vol. 60, No. 7, September 1981
Printed in U.S.A.

A Comparative Study of Several Dynamic Time-Warping Algorithms for Connected-Word Recognition

By C. S. MYERS and L. R. RABINER

(Manuscript received February 11, 1981)

Several different algorithms have been proposed for time registering a test pattern and a concatenated (isolated word) sequence of reference patterns for automatic connected-word recognition. These algorithms include the two-level, dynamic programming algorithm, the sampling approach and the level-building approach. In this paper, we discuss the theoretical differences and similarities among the various algorithms. An experimental comparison of these algorithms for a connected-digit recognition task is also given. The comparison shows that for typical applications, the level-building algorithm performs better than either the two-level DP*-matching or the sampling algorithm.*

I. INTRODUCTION

Research in the area of automatic speech recognition has progressed to the point at which a wide variety of isolated word recognition systems have been implemented and used successfully for many applications.[1-9] These systems have been used in such applications as data entry, searching, and sorting; however, use of these systems is restricted by the format of the speech input, i.e. isolated words. For many applications, a connected-word input format would have several advantages. Examples of such applications include:

(*i*) Credit card entry—A sequence of digits (and possibly letters) is required to specify the credit card number.

(*ii*) Directory listing retrieval—A sequence of letters is used to spell the name for which a directory listing is required.

(*iii*) Airline reservations—Sentences based on a restricted vocabulary and a restricted syntax are used to make reservations.

Several techniques for recognizing connected-word sequences from isolated word reference patterns have recently been proposed.[10–15]

In this paper, we compare, both theoretically and experimentally, three algorithms which have been proposed for connected-word recognition. These algorithms are the two-level, dynamic programming matching (TLDPM) approach, the sampling approach, and the level-building (LB) approach.[10–13] Another algorithm of the same general class as the ones considered here has been recently proposed by Bridle; however, it is not considered here.[14]

It should be noted that all of the algorithms which have been proposed for connected-word recognition are loosely related to a general information-theory-based algorithm proposed by Bahl and Jelinek.[15] However, each of these connected-word recognition algorithms make different assumptions, have different implementations, and make specific tradeoffs. Thus, the properties of these algorithms are entirely different from those of Bahl and Jelinek; therefore, they warrant independent study.

In Section II, we review each of the three connected-word recognition algorithms and then provide a theoretical comparison of the general properties. In Section III, we theoretically compare the connected-word recognition algorithms, in Section IV, we describe and give results of several experimental comparisons of these algorithms, and in Section V, we discuss these results and their implications for practical word-recognition systems.

II. CONNECTED-WORD RECOGNITION ALGORITHM

Figure 1 shows the basic structure for all the connected-word recognition algorithms under consideration. The feature extraction is generally similar to that used in most isolated word-recognition systems. Typical feature sets include energy of a set of bandpass filters,[16]

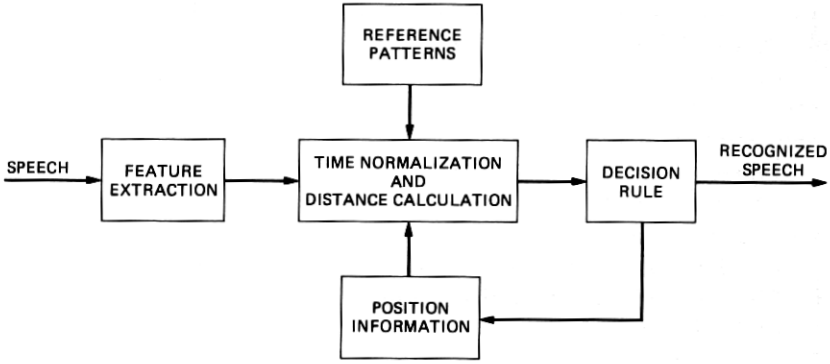

Fig. 1—Block diagram of a generic connected word recognition system.

and LPC coefficients.[14] Following feature extraction, the test utterance, now represented by a sequence of frames of the feature vector, is nonlinearly time-registered, with a set of reference patterns, and a set of distance, or dissimilarity, scores are calculated. Following time registration, a decision rule chooses the sequence of reference patterns which best matches the test utterance. Feedback, in the form of positional information, is used to determine which portion of the test utterance is to be matched to a given reference pattern. It is generally assumed that the test utterance consists of a sequence of words spoken by a *cooperative* user; that is, the words of the string are carefully articulated and spoken in a slow, deliberate manner, but not as isolated words. However, the reference patterns do consist of *isolated* words. The goal of the connected-word recognizer is to find the sequence of concatenated reference patterns, subject to given syntactical constraints, that best matches the test pattern. Among the issues which arise in solving for the best string are the following:

(*i*) How can the reference and test patterns be time-registered?

(*ii*) How can the reference patterns be modified to account for both coarticulation and the natural shortening of words inherent in connected speech?

(*iii*) Is the determination of the best concatenation of reference patterns done in a sequential manner, i.e. one decision at a time, or is some form of backtracking allowed?

(*iv*) How can alternative matches to the test utterance be generated in addition to the best match?

(*v*) Can syntactical constraints be used explicitly in the recognition stage or is some form of post-processing required?

In the following, we review the three connected-word recognition algorithms and discuss how each of them answers these questions.

2.1 Two-level DP-matching algorithm

The two-level DP-matching (TLDPM) algorithm[10] attempts to find the best concatenation of reference patterns to match a given test pattern (of length M frames) by first determining the optimal reference pattern to match any portion of the test pattern and then attempting to find the optimal way in which to concatenate these pieces. Figure 2a illustrates the first stage of this procedure. For all possible beginning points, b, and all possible references, an isolated word dynamic time warping (DTW) algorithm is used to find the best path to all of the possible ending frames, E_b. Distance scores for the best paths and the reference patterns which generates these best paths are recorded. The region over which a partial isolated word dynamic time warp is examined is shown in Fig. 2b. Here we show a region where the slope of the time warping function is restricted to be between ½ and 2 and

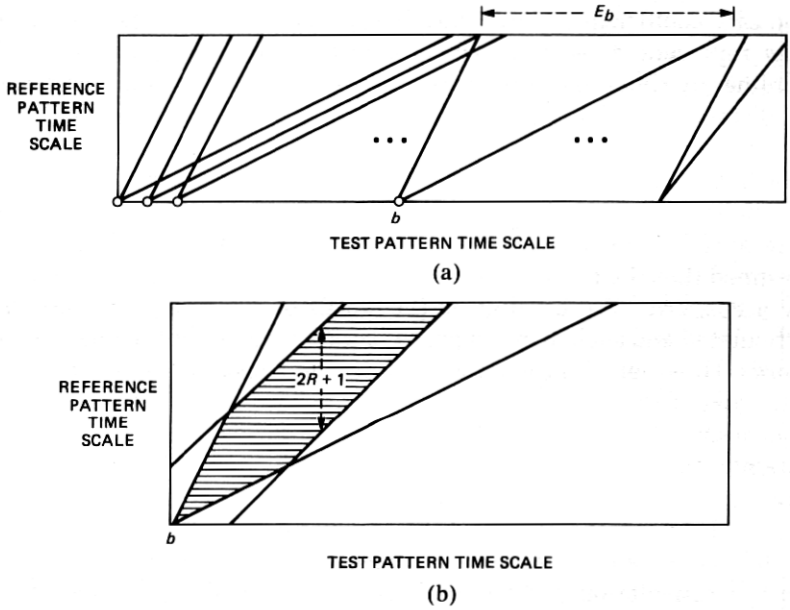

Fig. 2—Illustration of the TLDPM algorithm.

where the maximum amount that the time alignment contour may deviate from the line of slope one, which starts at the point $(b, 1)$, is plus or minus R frames.

If we denote the accumulated distance for the vth reference pattern from starting frame b to ending frame e as $\hat{D}(v, b, e)$, then the output of the first stage is the array of $\hat{D}$ values for all v, b, and e combinations. For any set of values of b and e, we can solve for the best reference pattern and distance as

$$D(b, e) = \min_{v} [\hat{D}(v, b, e)], \tag{1a}$$

$$N(b, e) = \operatorname*{argmin}_{v} [\hat{D}(v, b, e)], \tag{1b}$$

where $D(b, e)$ is the minimum accumulated distance from frames b to e of the test pattern, and $N(b, e)$ is the index of the reference pattern giving the minimum distance.

The second stage of the TLDPM algorithm is to determine the best match by piecing together the reference patterns in an optimal manner. This is accomplished again using a dynamic programming algorithm which finds the best concatenation of l reference patterns, ending at frame e of the test pattern by trying all concatenations of a portion of the test pattern ending at frame e and all best reference pattern concatenations of length $l - 1$, i.e.

$$\tilde{D}_l(e) = \min_{1 \le b < e} [D(b, e) + \tilde{D}_{(l-1)}(b - 1)] \tag{2a}$$

$$\tilde{D}_0(0) = 0, \tag{2b}$$

where $\tilde{D}_l(e)$ is the accumulated distance generated by matching the best concatenation of l reference patterns to the portion of the test pattern between frame 1 and frame e and where $D(b, e)$ is the distance associated with the best reference pattern used to match the portion of the test pattern between frame b and frame e. After computation of $\tilde{D}_l(e)$, the best string is recovered by first finding that value of l which minimizes $\tilde{D}_l(M)$ and then tracing back through the sequence of decisions which were used to generate $\tilde{D}_l(M)$.

The TLDPM algorithm made no inherent attempt to modify its reference patterns to account for either coarticulation or word shortening. While word shortening is, in general, compensated directly by the DTW, the use of reference patterns which are inherently longer than the test patterns to which they are to be matched makes this problem much more difficult. In addition, the lack of any boundary modifications for the reference patterns must inherently reduce the potential accuracy of the TLDPM approach.

It is obvious that the TLDPM approach is not a sequential decision method, since no partial match is firmly decided on until the entire second pass of the algorithm is completed. It should also be clear that it is possible to generate not only the best string but the K best strings by simply keeping track of the K best reference patterns which match any portion of the test pattern and also by keeping track of the K best strings for every step of the second pass. Finally, it is clear that, as described, the TLDPM algorithm cannot make use of syntactical constraints directly but must make use of them in a post-processing stage to choose among various candidate strings. That is, syntactical constraints cannot be used to restrict which words of the vocabulary are used for any given beginning and ending point pair, unlike the LB algorithm in which levels correspond to word positions in the string. However, once the beginning and ending point pair distance matrices have been generated, then syntactical constraints can be applied in the second stage of the TLDPM algorithm.

2.2 The sampling approach

Unlike the TLDPM algorithm, the sampling algorithm is more of a sequential decision process. The sampling approach attempts, via a local minimum DTW algorithm,[17] to match a reference pattern to a portion of the test pattern. Unlike the two-level approach of the preceding section, not all portions of the test pattern are tested. Instead, only a small subset of the test pattern is used. The way in which the regions of the test pattern are chosen is as follows. Following

the time registration of the reference pattern to one portion of the test pattern, an ending point is implicitly defined by that frame of the test pattern which best matches the end of the reference pattern. After all reference patterns have been tried, the one which gives the best match is chosen as the proper word and the ending point of the test pattern, associated with this reference, is used to hypothesize a beginning region within the test pattern for the next set of references. This procedure is illustrated in Fig. 3. Reference pattern number 1, which is hypothesized to begin somewhere within beginning region 1, is time-registered to the test pattern using a local minimum DTW algorithm. Once the ending point for the best match for Ref. 1 has been determined, a beginning region for the next reference pattern is determined. In general, this beginning region is centered somewhat earlier in the test pattern than the ending position of the previous word. This is done to compensate for the difference in durations between concatenated reference patterns and connected word utterances, and to account for coarticulation between words.

Note that a categorical decision as to the best reference need not be made if the distance scores indicate a high likelihood of confusion. In such cases, when the distance scores for two or more reference patterns are approximately equal, the sampling method keeps tract of all possible strings using the approach described above. Thus, the possi-

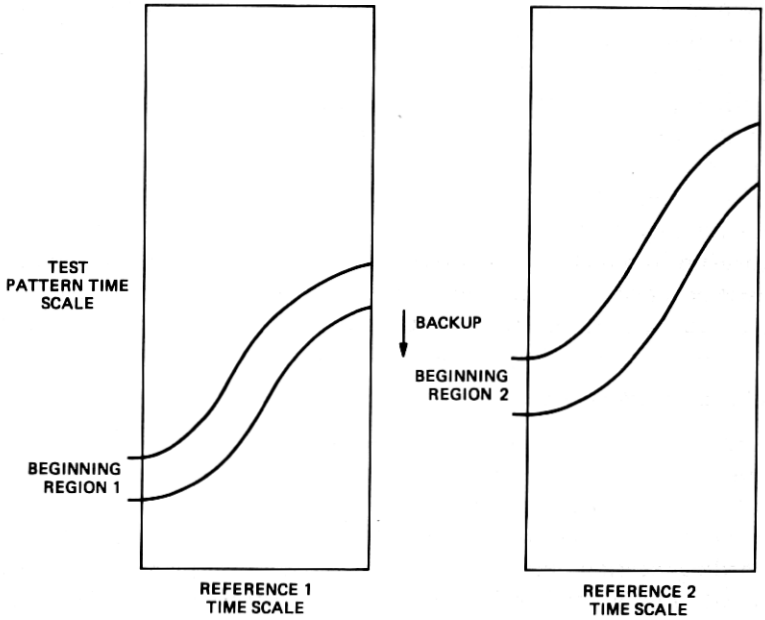

Fig. 3—Illustration of the sampling algorithm.

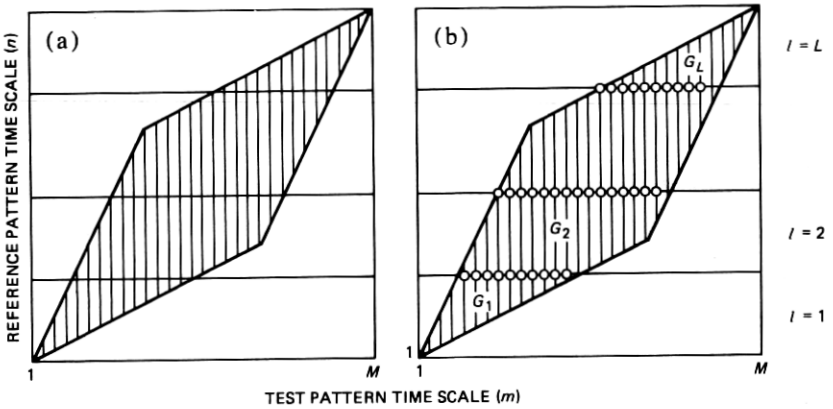

Fig. 4—Two possible implementations of a constrained DTW algorithm.

bility exists of finding a best string at the end of the search, which is different from the best string as the search proceeds from left to right.

In addition to effectively modifying reference patterns by overlapping beginning and ending regions, the sampling algorithm can also use the local minimum DTW algorithm in such a way as to eliminate part of the end of any reference pattern. This is done by not forcing the local minimum DTW algorithm to proceed all the way to the end of the reference pattern, but rather to stop at some frame before the end. Thus, the sampling approach has more inherent flexibility in dealing with modifications to the reference patterns than the TLDPM method.

In contrast to the TLDPM algorithm, the sampling algorithm is able to use syntactical constraints (in the form of a regular grammar) directly, rather than in a post-processing stage. Such a method is described in detail by Levinson and Rosenberg,[12] but the main idea is to trace the graph which represents the grammar simultaneously with matching the reference patterns within the test pattern by keeping track of not only the current state, but also the current ending frame within the test pattern.

2.3 The level-building algorithm

Figure 4 illustrates the basic idea involved in the LB algorithm. Here we show how a constrained endpoint DTW algorithm, in which the slope of the warping function is restricted to be between ½ and 2, is used to find the best alignment between a test pattern and a given concatenated sequence of L reference patterns. In Fig. 4a, we show the computation proceeding in a sequence of vertical strips. Figure 4b shows an alternative way in which the computation may be performed. A set of horizontal lines has been drawn to indicate the boundaries between the different reference patterns in the concatenated reference

pattern. We may now compute the optimal, time alignment path in successive levels by using the distances accumulated at the end of one level to initialize the next level. The LB algorithm uses this decomposition to find the optimal sequence of reference patterns by trying all possible reference patterns at any level and recording, for each ending frame at that level, the reference pattern which gave the best distance to that ending frame and a pointer back to the previous levels. These minimum distances are used to initialize the following level. After the final level has been examined, the optimal path is recovered by tracing back along the chain of pointers.

We have demonstrated that the algorithm solves exactly the same problem as the TLDPM algorithm.[13] In addition, we have shown how to modify the LB algorithm via a set of parameters to give it more flexibility. These parameters, as shown in Fig. 5, are as follows:

(*i*) δ_{R_1} —Region of uncertainty at the beginning of the reference pattern.

(*ii*) δ_{R_2} —Region of uncertainty at the end of the reference pattern.

(*iii*) δ_{END} —Region of uncertainty at the end of the test pattern.

(*iv*) M_T —Multiplier used to reduce the size of the beginning region for any level.

(*v*) ϵ —Parameter used to restrict the size of any vertical strip.

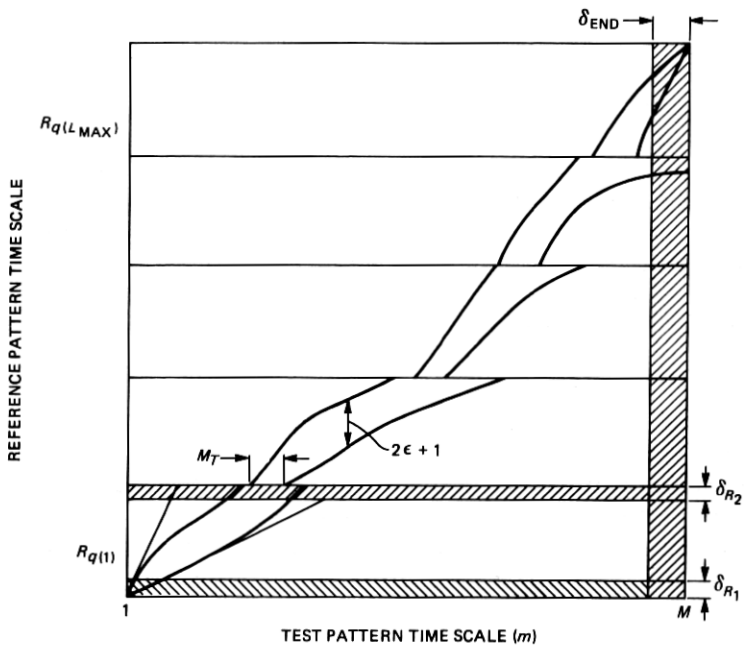

Fig. 5—Illustrations of the parameters of the LB algorithm.

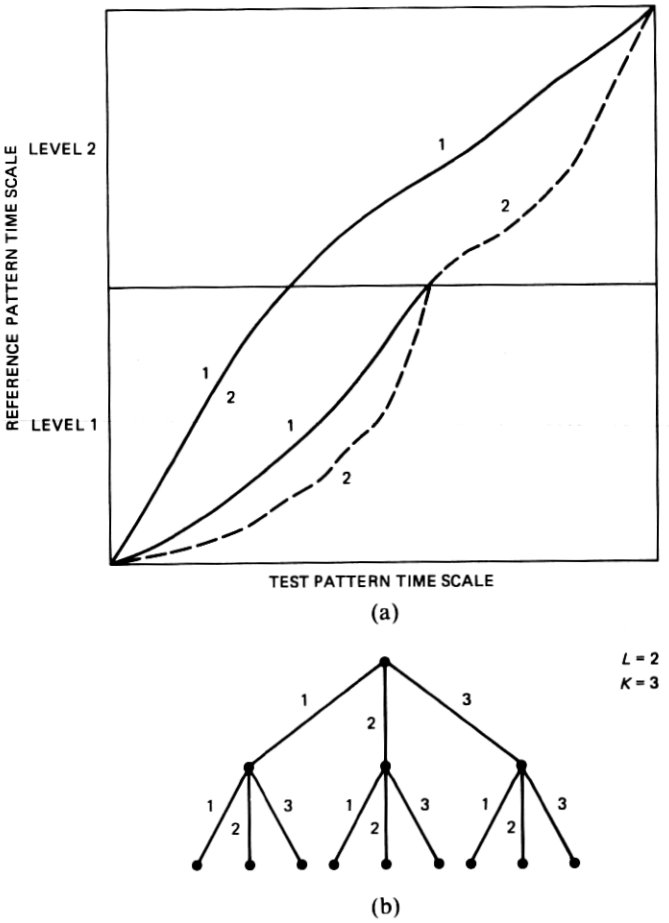

Fig. 6—Generation of multiple candidate strings for the LB algorithm.

The effects of these parameters are shown in Fig. 5. We observe that δ_{R_1} and δ_{R_2} define regions, at the beginning and end of each reference pattern, in which the local path may begin or end. In this manner, some of the gross features of coarticulation and length reduction present in a connected-word utterance may be accounted for. Thus, the LB algorithm has some of the flexibility inherent in the sampling algorithm.

In addition to modifying templates, the algorithm has the advantage of not being a sequential decision process and, thus, has the ability to recover from mistakes as in the TLDPM algorithm.

One important shortcoming of the LB algorithm is its ability to generate alternative candidates. The method by which multiple can-

didates are generated is to record not only the best candidate to each ending frame of each level, but to record the K best candidates and then to allow substitutions in the traceback. This method is illustrated in Fig. 6a for $K = 2$ candidates and for $L = 2$ levels. The solid paths represent the best paths from the beginning of the level to that particular ending point and the dashed paths represent the second best path to that particular ending point (using a different reference than the one used in the best path). We see that for $K = 2$ and $L = 2$ we may generate four different candidate strings and, in general, may have K^L different candidate strings. Such a situation may be represented by a tree with a branching factor of K and a depth of L as shown in Fig. 6b for $K = 3$, $L = 2$. Generation of the possible candidate strings is simply a tree-searching problem, and it is possible to reduce the amount of traceback by pruning the tree. To prune the tree we may take advantage of the fact that the kth best path at any node in the tree represents a string whose distance is always larger than the $(k - 1)$st best path to that node. The difficulty with such a scheme is that, unlike the TLDPM algorithm, we are not guaranteed of finding even the true second best path. Figure 7 illustrates this problem. Here the best path is given by string AA and an alternative is given by string CA. Another path, shown by string BA, must have a larger distance than string AA, but may have a smaller distance than string CA. However, the path BA will not be recorded because, at the second

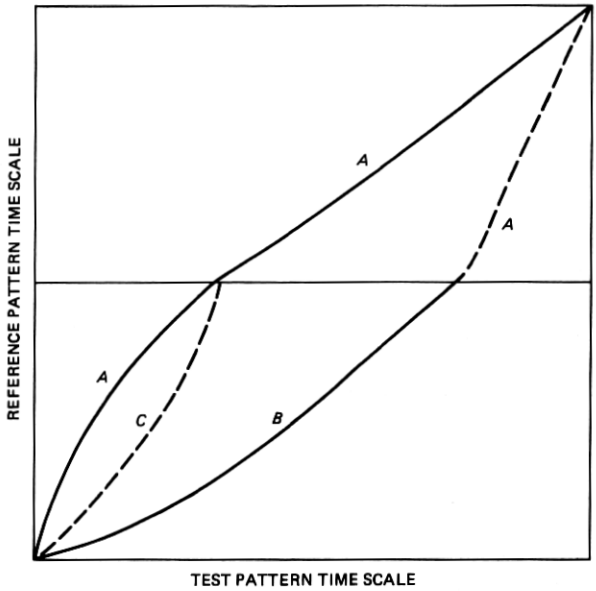

Fig. 7—Illustration of the failure of the LB algorithm to find all good candidate strings.

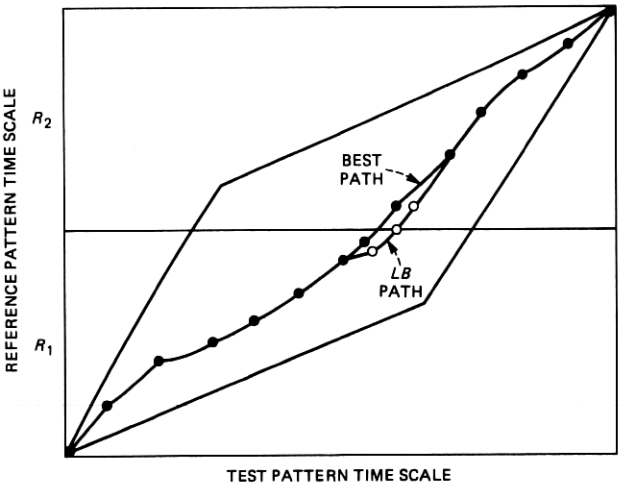

Fig. 8—Illustration of the path differences between the LB algorithm and a single time warp to the concatenated sequence of reference patterns.

level, only the *best* path using reference A is recorded. While it would be possible to record this second best path, the computational problems which would have to be overcome seem much too burdensome.

Another small theoretical difference between both the TLDPM and the LB solutions, and the exact best solution for the sequence of concatenated reference patterns that best matches the test pattern, is illustrated in Fig. 8. Here we show a sequence of two reference patterns, R_1 and R_2, and the best path for warping the super reference pattern formed by concatenating R_1 and R_2 to T, as well as the LB path. Since the LB path is constrained to end at the end of each reference pattern (level), then a path point is constrained to occur at the end of each reference pattern. This need not be the case for the optimum concatenation and, thus, a small difference can exist in the solutions. In practice, this effect does not occur (unless $\delta_{R_1} = \delta_{R_2} = 0$) since the δ_{R_1}, δ_{R_2} parameters allow paths to skip frames of the reference patterns.

A final consideration involving the LB algorithm is the use of syntactical constraints. Since all good candidates may not be generated by the LB algorithm, it is important that syntactical constraints be easily incorporated into its structure. Fortunately, this is possible and has been described in detail by Myers and Levinson.[18] The basic principle is to build up the results by states of a finite-state automata, rather than by levels, and to use transitions of the finite state automata to guide the recognition process.

In Section III, we summarize the qualitative features of the different connected-word recognition algorithms and also give a quantitative evaluation of both their space and time complexities.

III. THEORETICAL COMPARISON OF THE CONNECTED-WORD RECOGNITION ALGORITHMS

In Table I we summarize our qualitative evaluation of the different connected-word recognition algorithms. Both the sampling and LB algorithms have the ability to modify their reference patterns, although in different ways. The sampling approach allows both the removal of some frames from the reference pattern and the overlap of reference patterns, while the LB approach simply allows the removal of some frames from the reference patterns. Thus, we may conclude that the LB and the sampling algorithms will be able to more easily match test patterns which contain large amounts of coarticulation or length shortening than the TLDPM algorithm.

We have shown that the sampling algorithm is a sequential decision process, while both the TLDPM and the LB algorithms have some form of backtracking. Thus, we expect better performance from the TLDPM and the LB algorithms in those cases in which the sampling procedure may get lost.

In addition to being able to avoid getting lost, the TLDPM algorithm is the only algorithm that is capable of generating alternative candidates which are exactly correct. Thus, this algorithm should perform the best in cases in which many potential candidates are desirable. Unfortunately, as the final entry shows, the TLDPM algorithm is not well suited for syntactical analyses; therefore, it is more difficult to implement constraints using this algorithm.*

Table I—Qualitative comparison of connected-word recognition algorithms

Algorithm	Modification of References	Sequential Decision	Multiple Candidates	Syntax Constraints
Two-level DP-matching	No	No	Exact	No
Sampling	Yes	Yes	Heuristic	Yes
Level building	Yes	No	Heuristic	Yes

3.1 Computational comparison

In comparing both the time and space complexity of the three connected-word recognition algorithms, we shall assume that we are given a test utterance of length M frames, a vocabulary containing V words, a set of reference patterns with average length $\bar{N}$ (frames), with maximum time alignment deviation of $\pm R$ frames. For purpose of

* Note that it *is* possible to implement both syntactical constraints and reference pattern modifications directly in the TLDPM algorithm without changing the fundamental ideas contained in the algorithm. For purposes of our comparison, however, we have considered the TLDPM algorithm only as it was originally described.

comparison, our measure of time will be the number of times that a frame of the test utterance is compared to a frame of any reference pattern. For the TLDPM algorithm, the number of comparisons is given by

$$NC_{\text{TLDPM}} = V \cdot M \cdot \bar{N} \cdot (2R + 1), \tag{3}$$

since there is an isolated word time-warp of size $\bar{N} \cdot (2R + 1)$ for all V possible words, and for each of the M test frames.

For the sampling algorithm, the number of comparisons is given by

$$NC_s = V \cdot L \cdot \bar{N} \cdot (2\bar{\epsilon} + 1) \cdot \bar{\gamma}, \tag{4}$$

where L is the actual number of words in the test utterance, $\bar{\epsilon}$ is the range for the local minimum DTW algorithm, and $\bar{\gamma}$ is the average number of candidate strings which are retained. (Rabiner and Schmidt found that $\bar{\gamma}$ was, on average, between 1 and 2).[11] Data in this figure result from one local minimum time-warp of size $\bar{N} \cdot (2\bar{\epsilon} + 1)$ for each possible reference, for each word of the test pattern and for each candidate string.

For the LB algorithm, in which the slope of the warping function is restricted to be between ½ and 2 (as shown in Fig. 4), the number of comparisons is given by

$$NC_{\text{LB}} = V \cdot L_{\text{MAX}} \cdot M \cdot \bar{N}/3, \tag{5}$$

because the size of parallelogram in Fig. 4 is about $M \cdot \bar{N} \cdot L_{\text{MAX}}/3$, and because all V references must be used at each level.

Finally, by incorporating the range reduction techniques described by Myers and Rabiner,[13] the number of comparisons required for the reduced LB technique becomes

$$NC_{\text{LBR}} = V \cdot L_{\text{MAX}} \cdot \bar{N} \cdot (2\epsilon + 1), \tag{6}$$

since the reduced LB technique uses only a single local minimum time warp per level.

Table IIa summarizes the computational aspects of these connected-word DTW algorithms. The row-labelled number of basic time warps refers to the number of times that a basic DTW algorithm is applied. The size of the time warp is the average size of these basic time warps, and the total is the number of comparisons given in eqs. (3) to (6).

Table IIb gives a numerical comparison of the computation required by the various connected-word recognition algorithms for the case of

$$L_{\text{MAX}} = 5,\ V = 10,\ M = 120,\ \bar{N} = 35,$$

$$\bar{\epsilon} = 8,\ \epsilon = 12,\ R = 12,\ \bar{\gamma} = 1.5,\ L = 4.$$

Note that the total computation required by the TLDPM algorithm is 15 times that of the LB algorithm and 30 times that of either the

Table IIa—Computational comparisons of connected-word DTW algorithm

	Level Building	Two-Level DP Matching	Sampling	Reduced Level Building
Number of basic time warps	$L_{MAX} \cdot V$	$M \cdot V$	$L \cdot V \cdot \bar{\gamma}$	$L \cdot V$
Size of time warps	$\bar{N} \cdot M/3$	$\bar{N} \cdot (2R + 1)$	$\bar{N} \cdot (2\bar{\epsilon} + 1)$	$\bar{N} \cdot (2\epsilon + 1)$
Total computation for distance	$L_{MAX} \cdot V \cdot \bar{N} \cdot M/3$	$M \cdot V \cdot \bar{N} \cdot (2R + 1)$	$L \cdot V \cdot \bar{\gamma} \cdot \bar{N}(2\bar{\epsilon} + 1)$	$L \cdot V \cdot \bar{N}(2\epsilon + 1)$
Storage	$3 \cdot M \cdot L_{MAX} \cdot K$	$2 \cdot M \cdot (2R + 1) \cdot K$	0	$3 \cdot M \cdot L_{MAX} \cdot K$

Table IIb—Typical computational requirements for the case $L_{MAX} = 5$, $V = 10$, $M = 120$, $\bar{N} = 35$, $\bar{\epsilon} = 8$, $\epsilon = 12$, $\gamma = 1.5$, $L = 4$, $K = 2$, $R = 12$

	Level Building	Two-Level DP Matching	Sampling	Reduced Level Building
Number of basic time warps	50	1200	60	40
Size of time warps	1400	875	595	875
Total computation for distances	70,000	1,050,000	35,700	35,000
Storage	3600	12000	0	3600

reduced LB or the sampling algorithm. The efficiency of the LB algorithm derives partly from the fact that optimal paths to a range of ending frames for a range of starting frames are found simultaneously.

3.2 Storage comparison

In comparing the storage required by the various connected-word recognition algorithms, we refer only to that storage which varies among the different algorithms. Hence, storage for reference patterns, and storage needed by all basic DTW algorithms is not considered here. The storage for the TLDPM algorithm is given by

$$S_{\mathrm{TLDPM}} = 2 \cdot M \cdot (2R + 1) \cdot K. \tag{7}$$

Storage is required for the matrices of the K best distances and their associated words, for each of the $M \cdot (2R + 1)$ pairs of beginning and ending frames.

The storage for the sampling algorithm is so small (less than 100) as to be negligible. The storage for the LB algorithm is given by

$$S_{\mathrm{LB}} = 3 \cdot M \cdot L_{\mathrm{MAX}} \cdot K \tag{8}$$

since, at the end of each level, it is necessary to store a distance, an associated word, and a pointer to the previous level for each of the K possible candidates. (Clearly, the factor M in eq. (8) can be substantially reduced if storage of distances, words, and back pointers is used only for finite distance ending frames.)

The storage requirements for the four algorithms are summarized in the last row of Table IIa, and a numerical example ($K = 2$) is given in Table IIb. Note that both the LB and the TLDPM algorithm require a significant amount of storage, but that the storage required by the LB algorithm is only a third of that required by the TLDPM algorithm.

We have seen that both the LB and the sampling algorithm are computationally simpler than the TLDPM, but that the TLDPM algorithm has the potential to generate better candidates for a given connected-word recognition task. Thus, certain trade-offs exist and some important questions must now be considered. Among them are the following:

(*i*) Is the ability to modify the reference patterns a useful feature for a connected-word recognition algorithm?

(*ii*) Is the ability to backtrack a useful feature, or is a sequential decision process sufficient?

(*iii*) Are the additional computational costs of the TLDPM worth the increased number of good candidate strings?

In Section IV we give the results of several experimental studies designed to answer these questions.

IV. EXPERIMENTAL COMPARISONS OF CONNECTED-WORD RECOGNITION ALGORITHMS

To answer the questions at the end of Section III, each of the three connected-word recognition algorithms was simulated and tested on a common data base of 480 connected digit strings. The data base consisted of 80 strings of variable length (from two to three connected digits) spoken by each of six speakers (three male, three female). The strings were carefully articulated, and spoken in a slow, deliberate manner. Care was taken to guarantee an equal number of occurrences of all length strings, (from two to five digits), and an equal number of occurrences of all digits within the strings. This data base was used previously to evaluate the sampling method and the LB approach.[11,19]

Since results had been obtained previously on the sampling and LB algorithms, the TLDPM algorithm was all that had to be simulated and tested. For consistency, the LPC feature set used in the sampling and LB systems was used in the TLDPM system as the basic analysis features. The only variable parameter in the basic two-level DP warp method is the time adjustment parameter R. Sakoe indicated that a value of $R = 0.4\,\bar{N}$ would be appropriate where $\bar{N}$ is the average reference length.[10] Since $\bar{N} = 40$, this indicated that a value of $R = 16$ was required. The first experiment varied R from 8 to 99 to see its effect on recognition accuracy for a speaker-trained system. The results of this experiment (i.e., string error rate versus R) are given in Fig. 9. The results shown indicate that the larger the value of R, the lower the error rate, and that a flattening on the error curve occurs for values of $R \geq 20$.

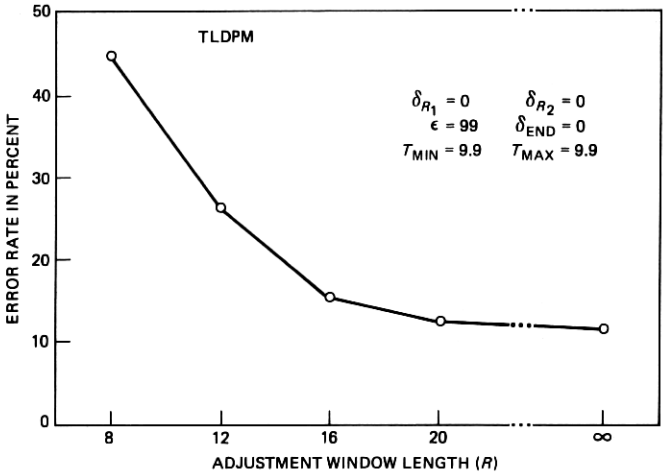

Fig. 9—String error rate as a function of the time shift parameter R of the TLDPM method for connecting digit strings.

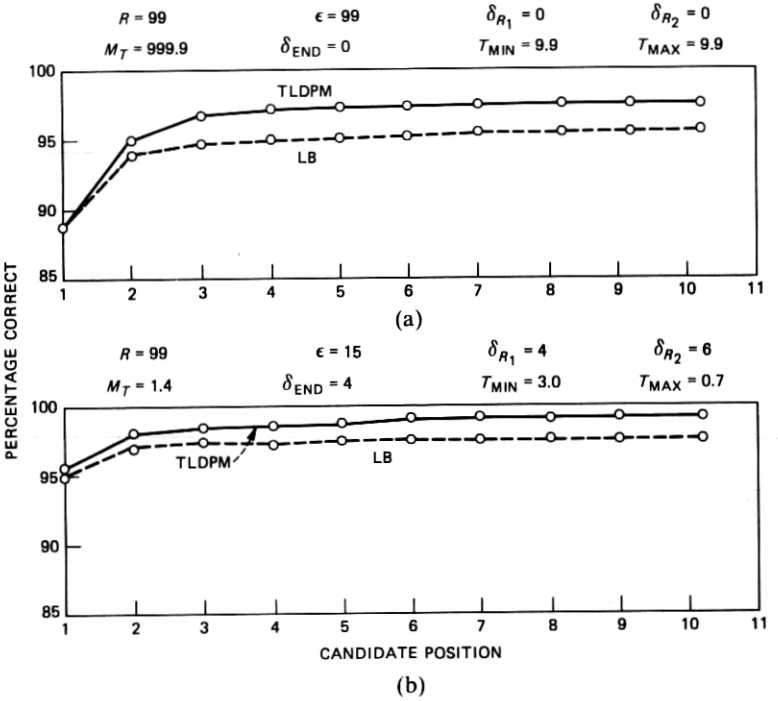

Fig. 10—Percentage correct strings as a function of candidate position for the TLDPM method and the LB method for (a) no reference modification and for (b) reference modification.

The next experiment made a direct comparison between the TLDPM method and the LB algorithm by setting the flexible LB parameters to values appropriate for the full range ($\epsilon = 99$, $M_T = 99.9$, $\delta_{END} = 0$, $T_{MIN} = 9.9$, $T_{MAX} = 9.9$, T_{MIN} and T_{MAX} specify distance thresholds used to reduce computation), and for no reduction in template lengths $\delta_{R_1} = \delta_{R_2} = 0$).[13] The $K = 2$ best candidates were recorded at each level and the 10 best strings (of any length) were found from the LB output. Similarly, for the two-level DP warp, the 10 best strings were found directly. Figure 10a shows the percentage of strings correct as a function of candidate position in the list for both the TLDPM method (solid curve) and the LB method (dashed curve). Again, a speaker-trained system was used. The results show the same string error rate (11.5 percent) on the top candidate for both methods as anticipated from the earlier discussion. However, the results also show a 1 to 2 percent higher string accuracy for the TLDPM method for candidate positions 2 to 10, indicating the potential improvements obtained by keeping distances for all posible beginning and ending frames.

Since the overall performance of the LB method was significantly better, using the best values of the LB parameters, rather than those shown in Fig. 10a, another experiment was run to compare recognition accuracy versus candidate position for both methods at the speaker-trained operating point ($R = 99$, $\epsilon = 99$, $M_T = 1.4$, $\delta_{END} = 4$, $T_{MIN} = 3.0$, $T_{MAX} = 0.7$, $\delta_{R_1} = 4$, $\delta_{R_2} = 6$). It should be noted that in this case, the TLDPM algorithm was modified from its original specifications (to a form similar to the UE2-1 algorithm[17]) to incorporate the parameters of the LB algorithm. These results are plotted in Fig. 10b. For the best candidate, the string error rates were 4.6 percent for the TLDPM method, and 4.8 percent for the LB method. For candidate positions 2–10, the TLDPM method has from 1 to 1.7 percent higher accuracy than the LB method. These results again indicate the small, but consistent improvement obtained by the TLDPM method, at least for alternative best string candidates. Whether or not this improved accuracy on higher candidate positions can be used depends strongly on the task, and the types of errors that occurred. Clearly, errors in string length can often be simply corrected, whereas errors of the same string length are generally difficult to detect and correct.

A summary of the resulting string error rates on the top recognition candidate for all three connected-word recognizers is given in Table III. For the sampling and LB methods, results are also given for performance in a speaker-independent mode (using 12 isolated speaker-independent templates per digit). No results are given for the TLDPM system as a speaker-independent recognizer because of the inordinate amount of running time (about 4 to 5 hours/string) required to evaluate its performance.

4.1 Performance of the LB method for other applications

The LB method of connected-word recognition was also applied to the problem of recognizing names from a given directory from spoken, spelled input. Since the vocabulary here is letters of the alphabet, and because of the high degree of confusability among several of the letters (e.g., *B*, *D*, *P*, *V*, etc.), a somewhat different structure was used for recognizing the names.[9] First, a name class is found, and then all names (if any) within the name class are tested and a name score is stored.

Table III—String error rates on connected digits

	Word Recognition Method			
	TLDPM (original)	TLDPM (modified)	Sampling	LB (reduced)
Speaker-trained	11.5%	4.6%	6.7%	4.8%
Speaker-independent	-	-	9.0%	4.6%

Table IV—Percentage names correct using the LB method

	Talking Speed	
	Deliberate	Normal
Speaker-trained	99%	90.5%
Speaker-independent	96%	87.5%

The name with the smallest name score is chosen as the recognized name.

To evaluate the performance of this system, four speakers (two male, two female) spoke 50 randomly selected names each as a connected sequence of letters of the last name, followed by a pause, followed by the initials. Each speaker spoke the name list two times. The first time, they spoke the names in a deliberate, carefully articulated manner, the second time, in a normal manner. The overall name accuracy achieved using the LB system is given in Table IV for both a speaker-trained and a speaker-independent implementation. Name accuracies of 96 to 99 percent were obtained for the deliberately spoken names, and of 87.5 to 90.5 percent for normally spoken names.

In addition to directory assistance, the LB algorithm has also been applied to the problem of sentence recognition for an airlines reservation system.[18] In this experiment, four speakers (two male and two female) spoke 50 sentences each. These sentences were formed from a 127-word vocabulary using a regular grammar. The grammar consisted of 144 states and 450 transitions. The sentences were spoken in a normal manner. Recognition was performed in a speaker-dependent manner. Sentence accuracies of 89 percent and word accuracies of 94 percent were obtained. These experiments on directory assistance and sentence recognition demonstrate that the LB algorithm has widespread applicability.

V. DISCUSSION AND SUMMARY

The answers to two of the three questions at the end of Section III are clear. The gain in accuracy obtained by modifying the reference pattern is quite large for the connected-digit sequences (e.g., the string error rate falls from 11.5 percent to 4.6 percent for the TLDPM method). Similar improvements have previously been noted for the LB algorithm.[19] Thus, it is clear that the modified reference patterns lead to improved overall performance. However, the results on recognizing spelled names (see Table IV) indicate clearly that as the rate of talking goes up, the performance of the system becomes dramatically worse. At this point, the basic model assumptions begin to fall apart and no simple modification of the reference patterns is adequate.

The answer to the second question concerning the value of a backtracking versus a sequential approach can be obtained from Table III. By comparing performance of the sampling approach (with sequential decisions) to the LB or TLDPM results (with backtracking), we see a loss in string accuracy of 2 to 4.4 percent using the sequential approach. Since there is essentially no computational gain which accompanies this loss in accuracy, it seems clear that a backtracking approach is superior to a sequential decision method.

The final question in section IV—whether the improved accuracy in higher candidate positions of the TLDPM method justifies the greatly increased computational load—is hard to answer. Since the recognition accuracy on the best string is the same, and since a cost factor in computation of about 40-to-1 is incurred, it would appear that the increased accuracy is not sufficient to justify the increased computation. However, there may exist tasks with constrained syntax such that the improved accuracy does justify the costs.

The overall conclusion is that both the LB and TLDPM methods provide high accuracy in recognizing strings of connected words. Moreover, since the computation of the LB method is considerably less than the computation of the TLDPM, this method is to be preferred for most applications.

REFERENCES

1. T. B. Martin, "Practical Applications of Voice Input to Machines," Proc. IEEE, *64* (April 1976), pp. 487–501.
2. M. B. Herscher and R. B. Cox, "Source Data Entry Using Voice Input," Conf. Record 1976 IEEE Int. Conf. on Acoustics, Speech, and Signal Processing (April 1976), pp. 190–3.
3. S. L. Moshier, "Talker Independent Speech Recognition in Commercial Environments," Speech Comm. Papers, Acoustics Soc. of Amer., Paper YY12, J. J. Wolf and D. H. Klatt, Eds., June 1979, pp. 551–3.
4. F. Itakura, "Minimum Prediction Residual Principle Applied to Speech Recognition," IEEE Trans. on Acoustics, Speech, and Signal Processing, *ASSP-23*, No. 1 (February 1975), pp. 67–72.
5. A. E. Rosenberg and F. Itakura, "Evaluation of an Automatic Word Recognition System Over Dialed-Up Telephone Lines," J. Acoust. Soc. Am., *60*, Suppl. 1 (November 1976), pp. S12 (Abstract).
6. S. E. Levinson et al., "Interactive Clustering Techniques for Selecting Speaker-Independent Reference Templates for Isolated Word Recognition," IEEE Trans. on Acoustics, Speech, and Signal Processings, *ASSP-27*, No. 3 (April 1979), pp. 134–41.
7. L. R. Rabiner et al., "Speaker-Independent Recognition of Isolated Words Using Clustering Techniques," IEEE Trans. on Acoustics, Speech, and Signal Processing, *ASSP-27*, No. 4 (August 1979), pp. 236–349.
8. L. R. Rabiner, J. G. Wilpon, and A. E. Rosenberg, "A Voice Controlled Repertory Dialer System," B.S.T.J., *59* (September 1980), pp. 1153–63.
9. B. Aldefeld et al., "Automated Directory Listing Retrieval System Based on Isolated Word Recognition," IEEE Proc., *68*, No. 10 (November 1980), pp. 1364–79.
10. H. Sakoe, "Two Level DP-Matching-A Dynamic Programming Based Pattern Matching Algorithm for Connected Word Recognition," IEEE Trans. on Acoustics, Speech, and Signal Processing, *ASSP-27*, No. 6 (December 1979), pp. 588–95.
11. L. R. Rabiner and C. E. Schmidt, "Application of Dynamic Time Warping to Connected Digit Recognition," IEEE Trans. on Acoustics, Speech, and Signal Processing, *ASSP-28*, No. 4 (August 1980), pp. 377–88.

12. S. E. Levinson and A. E. Rosenberg, "A New System for Continuous Speech Recognition—Preliminary Results," Proc. Int. Conf. Acoustics, Speech, and Signal Processing (April 1979), pp. 239–44.
13. C. S. Myers and L. R. Rabiner, "A Level Building Dynamic Time Warping Algorithm for Connected Word Recognition," IEEE Trans. on Acoustics, Speech, and Signal Processing, *ASSP-29*, No. 2 (April, 1981), pp. 284–97.
14. J. S. Bridle and M. D. Brown, "Connected Word Recognition Using Whole Word Templates," Proc. of the Institute of Acoustics, Autumn Conf., 1979.
15. L. R. Bahl and F. Jelinek, "Decoding for Channels with Insertions, Deletions, and Substitutions with Applications to Speech Recognition," IEEE Trans. on Info. Theory, *IT-21*, No. 4 (July 1975), pp. 404–11.
16. H. F. Silverman and N. R. Dixon, "A Comparison of Several Speech-Spectra Classification Methods," IEEE Trans. on Acoustics, Speech, and Signal Processing, *ASSP-24*, No. 4 (August 1976), pp. 289–95.
17. L. R. Rabiner, A. E. Rosenberg, and S. E. Levinson, "Considerations in Dynamic Time Warping for Discrete Word Recognition." IEEE Trans. on Acoustics, Speech, and Signal Processing, *ASSP-26*, No. 6 (December 1978), pp. 575–82.
18. C. S. Myers, and S. E. Levinson, "Sentence Recognition Using a Level Building Dynamic Time Warping Algorithm," Proc. Int. Conf. Acoustics, Speech, and Signaling Processing, 1981, April 1, 1981, pp. 956–9.
19. C. S. Myers and L. R. Rabiner, "Connected Digit Recognition Using a Level Building DTW Algorithm," IEEE Trans. on Acoustics, Speech, and Signal Processing, *ASSP-29*, No. 3 (June 1981), pp. 351–63.