Un algorithme systolique pour la reconnaissance de mots connectes

Jean-Pierre Banâtre, Patrice Frison, Patrice Quinton

INRIA

CENTRE DE RENNES

IRISA

Institut National
de Recherche
en Informatique
et en Automatique

Domaine de Voluceau
Rocquencourt
B.P 105
78153 Le Chesnay Cedex
France
Tél. 954 90 20

Rapports de Recherche

N° 148

UN ALGORITHME SYSTOLIQUE POUR LA RECONNAISSANCE DE MOTS CONNECTÉS

Jean-Pierre BANATRE
Patrice FRISON
Patrice QUINTON

Juillet 1982

IRISA

PUBLICATIONS INTERNES

A SYSTOLIC ALGORITHM FOR CONNECTED WORD RECOGNITION

Jean-Pierre BANATRE
Patrice FRISON
Patrice QUINTON

Publication Interne n°169 - Mai 1982

INTRODUCTION.

These last few years, much work has been devoted to speech recognition. Among the different problems that have been investigated, only isolated word recognition has proven reliable and efficient enough for commercial applications. Continuous speech recognition or at least connected word recognition would certainly be a considerable improvement that may lead to new applications of voice man-machine communication.

Solutions to these two problems rely on dynamic time warping (DTW) techniques introduced by Sakoe (1) and extended in (2, 3, 4) in order to cope with connected word recognition.

Recently, Ackland, Weste and Burr (5) and Banâtre, Frison and Quinton (6) proposed independently similar parallel algorithms for implementing efficiently DTW techniques. Ackland et al. (5) designed a network of processors for implementing classical DTW algorithms and built a prototype VLSI chip. Banâtre et al. proposed a similar structure for implementing a word-spotting algorithm based on Bahl and Jelinek statistical decoder (7). Performance estimation of these special purpose devices shows that it can improve by a factor of about 200 the time performance of conventional implementations.

The purpose of this paper is the description of a parallel algorithm for connected word recognition. This algorithm makes intensive use of pipelining and multiprocessing, it may be qualified of "systolic" (8). Its main properties can be summarized as follows :

(i) it solves the connected word recognition problem as stated by Sakoe (2) and Myers and Rabiner (4),

(ii) it is suitable for VLSI implementation,

(iii) it is a natural peripheral of the network presented by Ackland et al. (5), or with slight modification of the one of Banâtre et al. (6).

In the following, we give an overview of the main characteristics of the DTW algorithm for connected word recognition. A systolic implementation of this algorithm is then presented, and the main properties of the network presented in (5) and (6) are recalled. Then it is explained how this network can be connected to the machine presented in this paper.

CONNECTED DTW ALGORITHM.

Let T be a test string consisting of M frames. We denote by T(m) the vector of features of frame m and by T(i:j) (i≤j) the substring of T containing frames T(i), ..., T(j).

Reference patterns are denoted R_v, v∈[1,V], $N_v = |R_v|$ denotes the number of frames of R_v.

The connected word DTW algorithm aims in finding a sequence $R^S = R_{q(1)} \cdot R_{q(2)} \cdots R_{q(L)}$ consisting in the concatenation of L reference patterns such that DTW distance between R^S and T be minimal. Let us denote by $D(R^S,T)$ this distance and by $||R^S||$ the number of words of R^S. The problem we have to solve consists in evaluating the value D* defined as :

$$D^* = \min_{R^S} D(R^S,T) \tag{1}$$

Let D^*_L be the minimal distance between a super-reference of L words and T, we then have :

$$D^* = \min_L [D^*_L] \tag{2}$$

If the DTW algorithm uses asymetric slope constraints (2), D^*_L can be evaluated as :

$$D^*_L = \min_{q(1)\ldots q(L)} D(R_{q(1)}\cdots R_{q(L)},T)$$

$$= \min_{q(1)\ldots q(L)} [\min_B \sum_{i=1}^{L} D(R_{q(i)}, T(b(i) : e(i)))] \tag{3}$$

where $B = \{(b(i), e(i))\}_{i=1,L}$ is a sequence of indexes such that b(1) = 1, e(L) = M and ∀i : 1≤i≤L, b(i+1) = e(i)+1.

By reversing min's we get

$$D^*_L = \min_{(b(i),e(i))_{i=1,L}} [\sum_{i=1}^{L} \min_v [D(R_v, T(b(i) : e(i))]] \tag{4}$$

Let us set $\hat{D}(b,e) = \min_v D(R_v, T(b : e))$ (5)

and $D^*(e) = \min_{R^S} [D(R^S, T(1 : e)]$ (6)

and $D^*_L(e) = \min_{||R^S||=L} [D(R^S, T(1 : e)]$ (7)

(4) may also be formulated as

$$D^*_L(e) = \min_{1 \leq b \leq e} [D^*_{L-1}(b-1) + \hat{D}(b,e)] \tag{8}$$

and thus

$$D^*(e) = \min_L [\min_{1 \leq b \leq e} [D^*_{L-1}(b-1) + \hat{D}(b,e)]] \tag{9}$$

Reversing min's again produces

$$D^*(e) = \min_{1 \leq b \leq e} [\min_L D^*_L(b-1) + \hat{D}(b,e)] \tag{10}$$

One can observe that :

$$D^*(e) = \min_{1 \leq L \leq e} D^*_L(e) \tag{11}$$

since any reference pattern needs to be matched against at least one frame of T.

Finally (10) becomes :

$$D^*(e) = \min_{1 \leq b \leq e} [\min_{1 \leq L \leq b-1} D^*_L(b-1) + \hat{D}(b,e)] \tag{12}$$

or

$$D^*(e) = \min_{1 \leq b \leq e} [D^*(b-1) + \hat{D}(b,e)] \tag{13}$$

$D^*(e)$ can be computed in exactly e steps with the following iterative scheme :

$$\forall e > 0 \begin{cases} U_o(e) = +\infty \\ \\ U_k(e) = \min[U_{k-1}(e),\ D^*(k-1) + \hat{D}(k,e)] \end{cases} \tag{14}$$

with $D^*(0) = 0$. It is then clear that $D^*(e) = U_e(e)$ $\forall e \geq 0$. Although not detailed here, the last word of word string achieving $D^*(e)$ may be obtained during the iteration and would suffice to retrieve the entire word string by a backward linking mechanism.

In a first approximation, it may be noticed that the terms $U_k(e)$ with k fixed and $e \in [0,M]$ may be computed simultaneously - assuming of course that every term whose processing is completed be immediately available for further calculation. For example, as soon as $U_1(1)$ is evaluated it may be used for computation of terms $U_2(e)$, $e \in [2,M]$. Figure 1 illustrates dependencies between terms $U_k(e)$, $e \in [0,3]$.

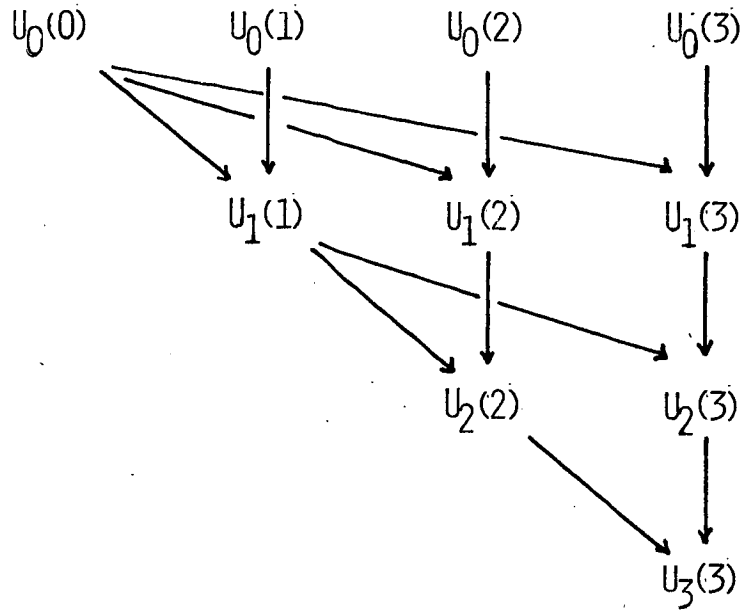

Figure 1 : Dependency graph for the calculation of terms $U_k(e)$ $(0 \leq e \leq 3)$.

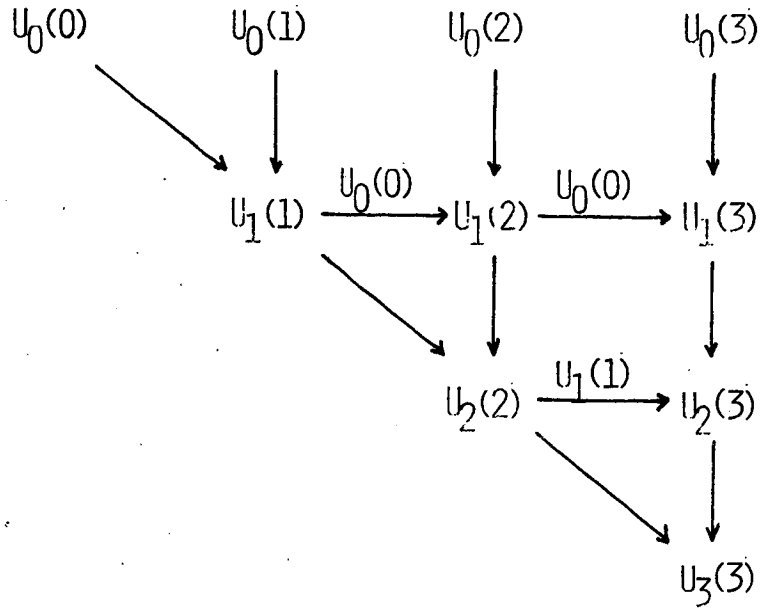

Figure 2 : Computations flow for terms $U_k(e)$ $(0 \leq e \leq 3)$.

However, it one thinks of designing a highly parallel algorithm (suitable for VLSI implementation), the above scheme cannot be used directly. It would necessitate that every $U_i(i)$ be connected to all $U_{i+1}(j)$, $j\varepsilon[0,M]$ and on a hardware point of view this is purely unrealistic. The next section presents a more reasonable scheme.

A SYSTOLIC IMPLEMENTATION OF THE ALGORITHM.

An essential feature of systolic "machines" is that they are all built from identical processors (8) that are connected in a regular fashion.

For our problem, a possible approach consists in organizing the parallel computation in such a way that long distance connection between processors be replaced by a succession of direct close connections - For example, instead of connecting $U_1(1)$ to $U_2(3)$ directly, we connect it to $U_2(2)$ and $U_2(2)$ to $U_2(3)$ - and information flow from $U_1(1)$ to $U_2(3)$ will follow this path (succession of elementary paths). According to this observation, the computation flow is exhibited on figure 2. It turns out that this computation may be performed on a linear array (D*-array) of processors conveniently connected.

Every cell of this array is structured as displayed in figure 3. It contains an adder and a comparator. The cell computes $U_k(e)$ using values $U_{k-1}(e)$, $U_{k-1}(k-1)$ and $\hat{D}(k,e)$.

Computation of $U_k(e)$ is achieved in two cycles (referred to as A and B). During cycle A, the cell inputs $\hat{D}(k,e)$ ($U_{k-1}(e)$ and $U_{k-1}(k-1)$ are assumed to be available). The cell computes $U_k(e)$ which is sent to its left neighbour. During cycle B, the cell reads $U_{k'-1}(e)$ and $U_{k'-1}(k'-1)$ to be used in the next A cycle. Figure 4 shows the overall organization of our network composed of M cells . Figure 5 describes steps of computation for $U_o(i)$, $U_1(i)$, $U_2(i)$ and $U_3(i)$, $i\varepsilon[0,3]$. Cell i computes $U_k(k+j-1)$, $k\varepsilon[0,M]$ and consequently values $U_k(k)$ are delivered by cell 1 after k executions of cycles A and B.

The next section describes how this network (called D*-array) may be connected to a device computing the values $\hat{D}(k,e)$ in order to solve the entire connected word recognition problem.

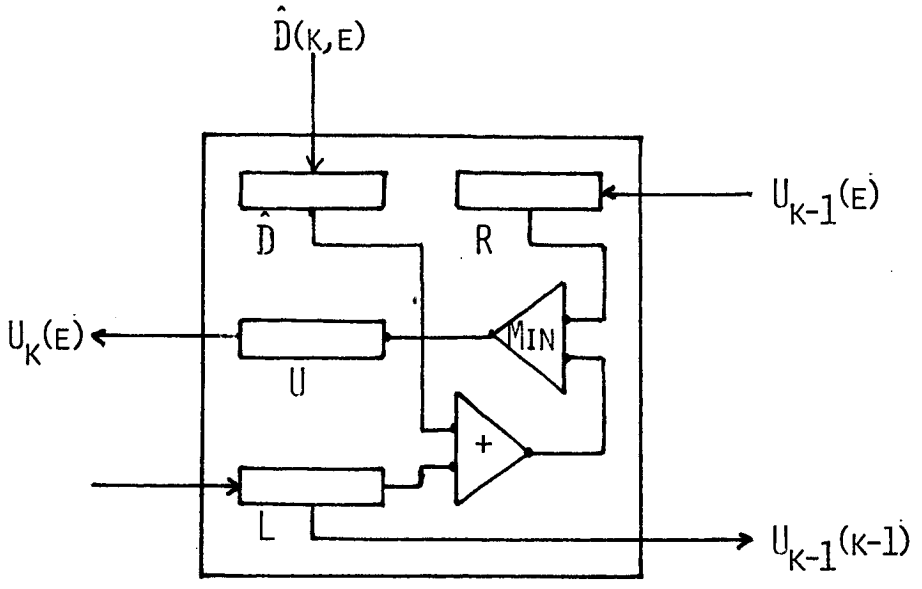

Figure 3 : Elementary cell structure.

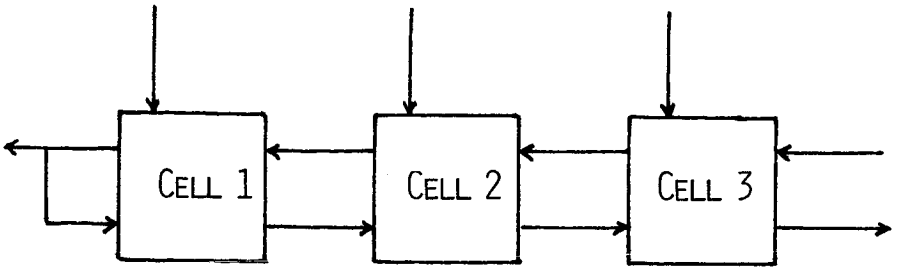

Figure 4 : Organization of the D*-array.

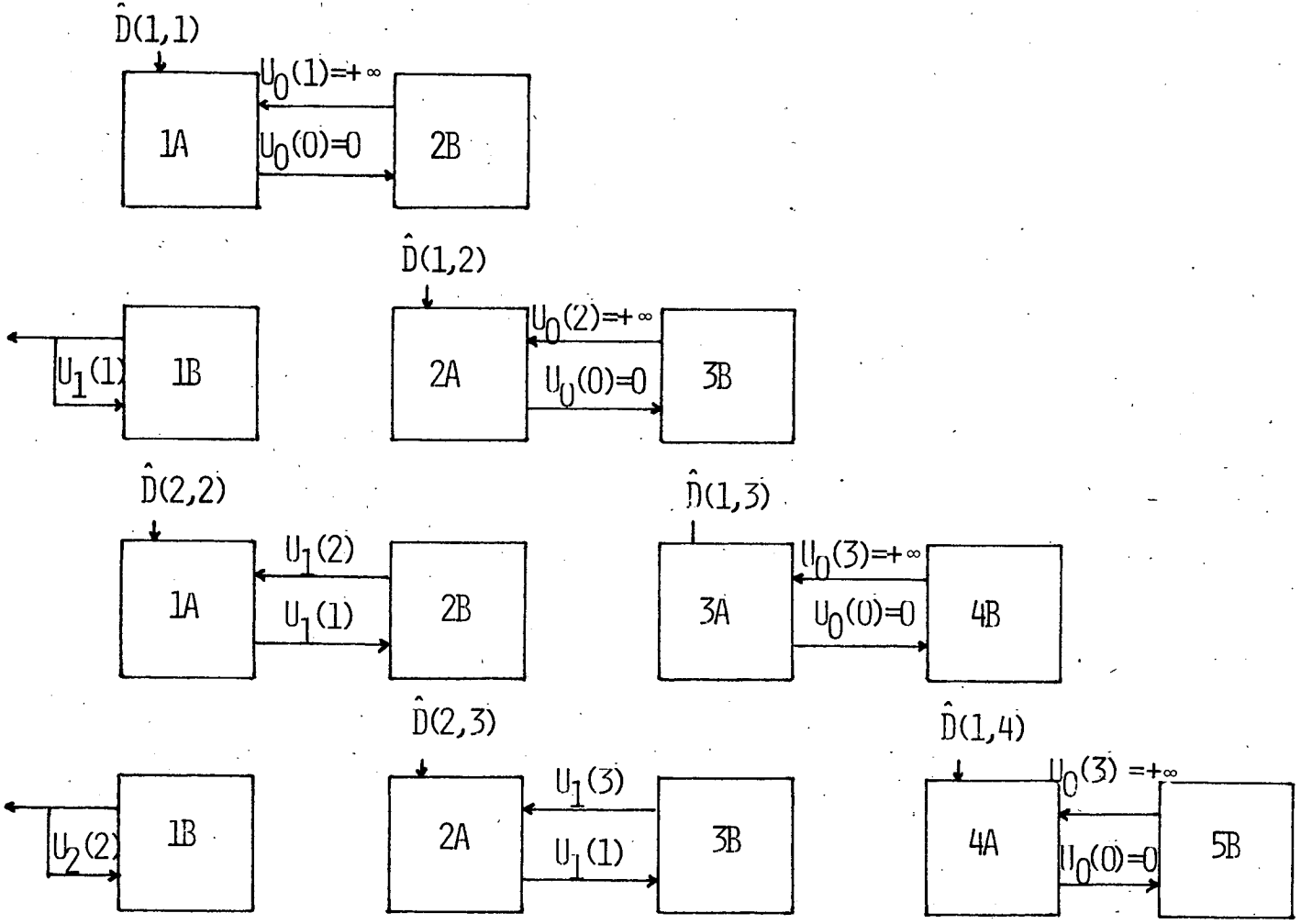

Figure 5 : State of the network at successive steps of the computation. A and B reffer to the cycle entered by processors.

A SYSTOLIC MACHINE FOR CONNECTED WORD DTW.

The whole machine is composed of three parts : the DTW network, the $\hat{D}$-array and the D*-array (figure 6).

The DTW network.

For clarity sake, it will be assumed that the computation of DTW distance is based on the simplest asymetric equation :

$$D(i,j) = \min\{D(i-1,j-1) + d(i,j),\ D(i-1,j) + d(i,j)\}$$
$$D(1,1) = d(1,1) \qquad (15)$$

where d(i,j) represents the distance between frame i of reference pattern and frame j of test pattern. Although this assumption is very unrealistic, it is demonstrated in (5) how more sophisticated slope constraints can be achieved using the same architecture.

The DTW network is composed of MxN* cells numbered <m,n>, $n\varepsilon[1,N^*]$, $m\varepsilon[1,M]$ where N* is the maximum reference pattern length.

The DTW network is vertically fed with a test pattern and horizontally (from left) with reference pattern.

Assume T(k) T(k+1) ... T(k+M) are vertical input and $R_v(1)$, $R_v(2)$, ..., $R_v(N^*)$ are horizontal inputs. Let us set $D(k,j,v) = D(R_v, T(k:j))$. Values D(k,j,v) are produced by processors <N*,j>. Computations are made on a diagonal basis, so that, if we assume that the computation starts at time t,

- T(k+j) enters cell <1,j> at time t+j-1
- $R_v(i)$ enters cell <i,1> at time t+i-1
- D(k,j,v) is produced by cell <N*,j> at time t+N +j-2 (see figure 7).

The comparison between V reference patterns and successive substrings T(k:k+j) of test pattern T are achieved by the following algorithm :

Step 1 : T(1:M) is input vertically. Reference patterns R_1, R_2, ..., R_V are pipelined horizontally. Processors <N*,j> output every value D(1,j,v) for $1 \leqslant v \leqslant V$;

Step k : T(k:M) is input vertically and reference patterns R_1 ... R_V are pipelined. Processors <N*,j> output D(k,k+j,v) for $1 \leqslant v \leqslant V$.

Call S the time when <N*,1> outputs the first result D(1,1,v). It is clear that <N*,1> will produce the sequence $\{D(1,1,v) \mid v=1, \ldots, V\}$ then $\{D(2,2,v) \mid v=1, \ldots, V\}$ and so on ... More generally, processor <N*,j> produces $\{D(1,j,v) \mid v=1, \ldots, V\}$ then $\{D(2,j+1,v) \mid 1 \leqslant v \leqslant V\}$ ... $\{D(k,j+k,v) \mid 1 \leqslant v \leqslant V\}$, starting at time S+j-1.

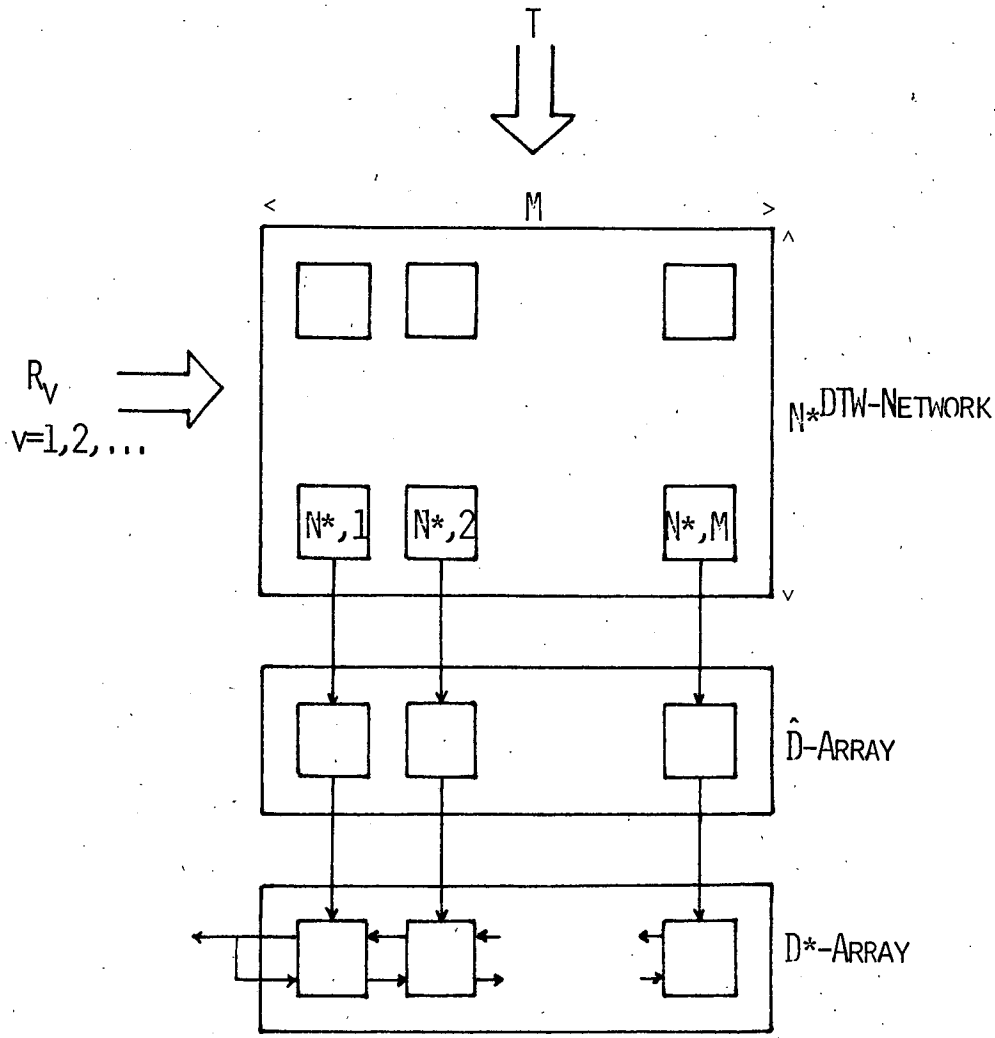

Figure 6 : Overall structure of the connected word DTW machine.

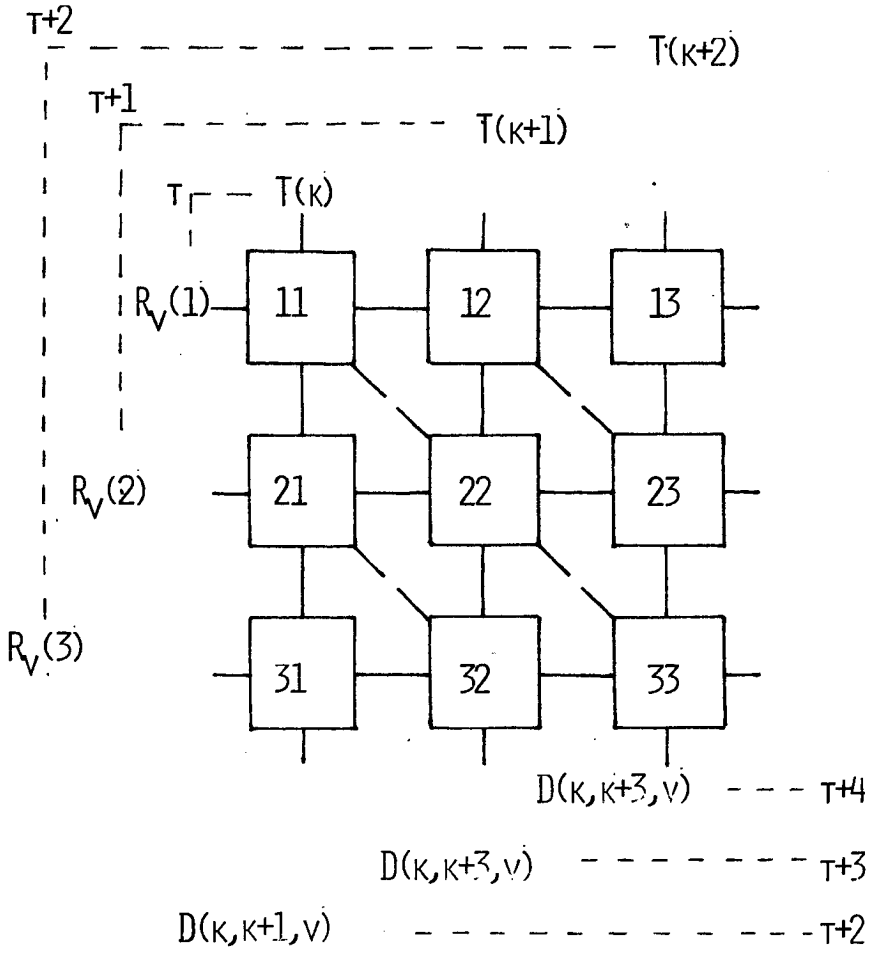

Figure 7 : Organisation of the DTW machine.

The $\hat{D}$-array.

This array computes the values $\hat{D}(i,j)$ consumed by the D*-array. $\hat{D}$-array is simply made out of M cells that read the values D(i,j,v) output by processors <N*,j> of the DTW-network and compute the value :

$$\hat{D}(i,j) = \min_{v} D(i,j,v)$$

It can be seen that cell 1 of the $\hat{D}$-array produces $\hat{D}(1,1)$ at time S+V-1, then $\hat{D}(2,2)$ at S+2V-1 ... $\hat{D}(k,k)$ at time S+kV-1. More generally, cell j produces $\hat{D}(k,k+j)$ at time S+j+kV-2.

Synchronization of the D*-array.

As we assumed that the basic cycles of every type of processor of the machine have the same duration, it is necessary to slightly modify the functioning of the D*-array. Let us consider the functioning of cell j of the D*-array, which has to compute $U_k(j+k)$, $k\varepsilon[0,M]$. Cell j may start to compute $U_k(j+k)$ as soon as $\hat{D}(k,j+k)$ is available, that is at time S+j+kV-2. At this time, $U_{k-1}(j+k)$ has been computed by cell j+1 and also $U_{k-1}(k-1)$ is available. Cell j thus may enter cycle A and cycle B and has to wait until $\hat{D}(k+1,j+k+1)$ is available at time S+j+(k+1)V-2. Consequently, cell j enters V-2 idle cycles. This means that V must be greater than 2 which is a reasonable assumption.

DISCUSSION.

As it has been described, the network comprises MxN* DTW-cells, M $\hat{D}$ and D* cells. It is well known that DTW computation need only to be made for (i,j) cells such that $|i-j| \leq r$. This remark reduces notably the size of the machine. On the other hand, it is also possible to use the same structure for word-spotting of phonetic transcription of words against the phonetic labelling of speech (Banâtre et al. (6)). The advantage of such an approach is that it reduces considerably the size of the network. However, it remains to be proven that, despite the uncertainty of phonetic labelling, recognition rate remains high enough in order to meet the requirements of CWR application.

REFERENCES.

(1) H. Sakoe, S. Chiba, "Dynamic programming algorithm optimization for spoken word recognition", IEEE Trans. ASSP, ASSP-26, Feb. 1978.

(2) H. Sakoe, "Two-level DP-matching. A dynamic programming based pattern matching algorithm for connected word recognition", IEEE Trans. ASSP, ASSP-27, Dec. 1979.

(3) L.R. Rabiner, C.E. Schmidt, "Application of dynamic time warping to connected digit recognition", IEEE Trans. ASSP, ASSP-28, Aug. 1980.

(4) C.S. Myers, L.R. Rabiner, "A level building time warping algorithm for connected word recognition", IEEE Trans. ASSP, ASSP-29, April 1981.

(5) B. Ackland, N. Weste, D.J. Burr, "An integrated multiprocessing array for time warp pattern matching", Sigarch Newsletter, vol. 9, n° 3, May 1981 (8th Annual symp. on Computer Architecture).

(6) J.P. Banâtre, P. Frison, P. Quinton, "A network for the detection of words in continuous speech", VLSI Int. Conf. (J.P. Gray Ed.), Academic Press, 1981.

(7) L.R. Bahl, F. Jelinek, "Decoding for channels with insertions, deletions and substitutions with application to speech recognition", IEEE Trans. Information Theory, IT-21, July 1976.

(8) H.T. Kung, "Let's design algorithms for VLSI systems", Proc. of the Caltech Conference on VLSI, Jan. 1979.

Liste des Publications Internes IRISA

PI 142 **Un lemme général de stabilité pour la commande adaptative en déterministe de systèmes non nécessairement à minimum de phase]:**
Cl. Samson , 40 pages ; *Novembre 1980*

PI 143 **Détection, Estimation de l'orientation et saisie d'une cible mobile par proximétrie optique**
B. Espiau , 142 pages ; *Janvier 1981*

PI 144 **Une contribution à l'étude de l'impact de l'informatique sur les organisations**
L. Breton, A. Prod'homme, J. Villard , 58 pages ; *Décembre 1980*

PI 145 **Rupture de modèles statistiques**
M. Basseville, A. Benveniste , 130 pages ; *Mars 1981*

PI 146 **Traitement des questionnaires avec non réponse, analyse des correspondances avec marge modifiée et analyse multicanonique avec contrainte**
B. Escoffier , 38 pages ; *Mars 1981*

PI 147 **Deux files d'attente à capacité limitée en tandem**
J. Pellaumail, J. Boyer , 19 pages ; *Juillet 1981*

PI 148 **Programme de classification hiérarchique : 1) Méthode de la vraisemblance des liens, 2) Méthode de la variance expliquée**
I.C. Lerman , 113 pages ; *Juin 1981*

PI 149 **Convergence des méthodes de commande adaptative en présence de perturbations aléatoires**
J.J. Fuchs , 46 pages ; *Juillet 1981*

PI 150 **Construction automatique et évaluation d'un graphe d'«implication» issu de données binaires, dans le cadre de la didactique des mathématiques**
H. Rostam , 112 pages ; *Juin 1981*

PI 151 **Réalisation d'un outil d'évaluation de mécanismes de détection de pannes]-]Projet Pilote SURF**
B. Decouty, G. Michel, C. Wagner, Y. Crouzet , 59 pages ; *Juillet 1981*

PI 152 **Règle maximale**
J. Pellaumail , 18 pages ; *Septembre 1981*

PI 153 **Corrélation partielle dans le cas « qualitatif »**
I.C. Lerman , 125 pages ; *Octobre 1981*

PI 154 **Stability analysis of adaptively controlled not-necessarily minimum phase systems with disturbances**
Cl. Samson , 40 pages ; *Octobre 1981*

PI 155 **Analyses d'opinions d'instituteurs à l'égard de l'appropriation des nombres naturels par les élèves de cycle préparatoire**
R. Gras , 37 pages ; *Octobre 1981*

PI 156 **Récursion induction principle revisited**
G. Boudol, L. Kott , 49 pages ; *Décembre 1981*

PI 157 **Loi d'une variable aléatoire à valeur $\mathbf{R}^+$** réalisant le minimum des moments d'ordre supérieur à deux lorsque les deux premiers sont fixés
M.Kowalowka, R. Marie , 8 pages ; *Décembre 1981*

PI 158 **Réalisations stochastiques de signaux non stationnaires, et identification sur un seul échantillon**
A. Benveniste J.J. Fuchs , 33 pages ; *Mars 1982*

PI 159 **Méthode d'interprétation d'une classification hiérarchique d'attributs-modalites pour l'«explication» d'une variable ; application à la recherche de seuil critique de la tension artérielle systolique et des indicateurs de risque cardiovasculaire**
B. Tallur , 34 pages ; *Janvier 1982*

PI 160 **Probabilite stationnaire d'un réseau de files d'attente multiclasse à serveur central et à routages dépendant de l'état**
L.M. Le Ny , 18 pages ; *Janvier 1982*

PI 161 **Détection séqentielle de changements brusques des caractéristiques spectrales d'un signal numérique**
M. Basseville, A. Benveniste , pages ; *Mars 1982*

PI 162 **Actes regroupés des journées de Classification de Toulouse (Mai 1980), et de Nancy (Juin 1981)**
I.C. Lerman , 304 pages ;

PI 163 **Modélisation et Identification des caractéristiques d'une structure vibratoire : un problème de réalisation stochastique d'un grand système non stationnaire**
M. Prévosto, A. Benveniste, B. Barnouin , 46 pages ; *Mars 182*

PI 164 **An enlarged definition and complete axiomatization of observational congruence of finite processes**
Ph. Darondeau , 45 pages ; *Avril 1982*

PI 165 **Accès vidéotex à une banque de données médicales**
A. Chauffaut, M. Dragone, R. Rivoire, J.M. Roger , 25 pages ; *Mai 1982*

PI 166 **Comparaison de groupes de variables définies sur le même ensemble d'individus**
B. Escofier, J. Pages , 115 pages ; *Mai 1982*

PI 167 **Transport en circuits virtuels internes sur réseau local et connexion Transpac**
M. Tournois, R. Trépos , 90 pages ; *Mai 1982*

PI 168 **Impact de l'intégration sur le traitement automatique de la parole**
P. Quinton , 14 pages ; *Mai 1982*

PI 169 **A systolic algorithm for connected word recognition**
J.P. Banâtre, P. Frison, P. Quinton , 13 pages ; *Mai 1982*

PI 170 **A network for the detection of words in continuous speech**
J.P. Banâtre, P. Frison, P. Quinton . 24 pages ; *Mai 1982*