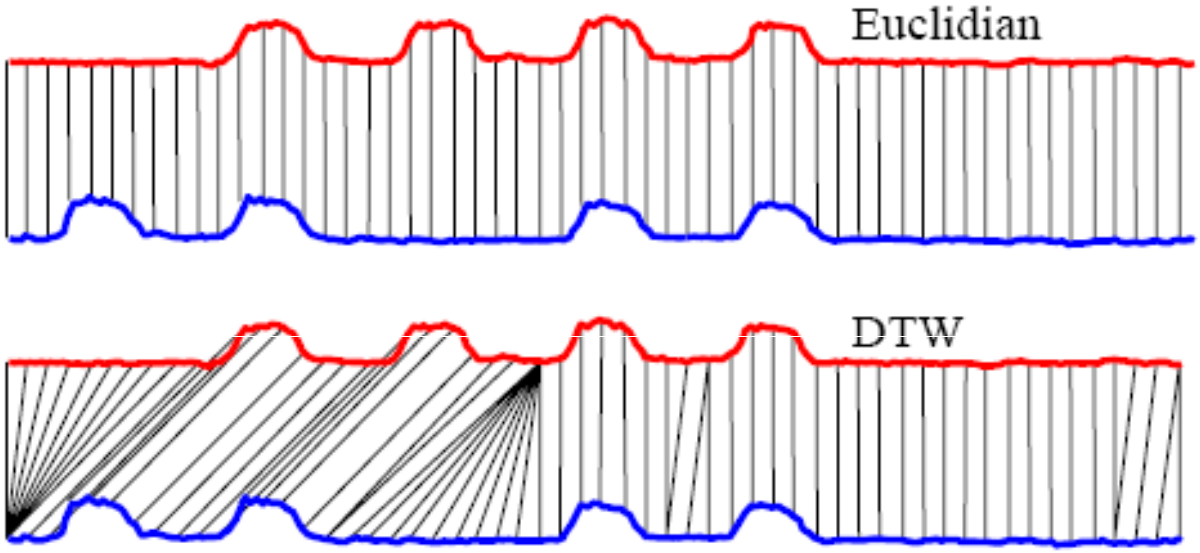
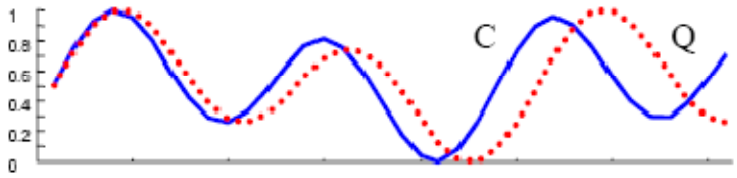
Dynamic Time Warping

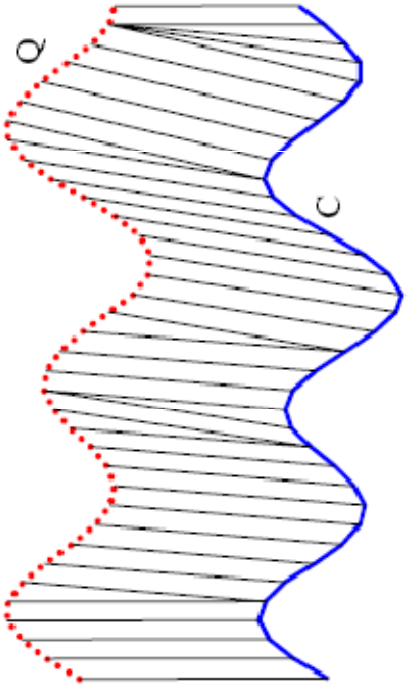
George Darmiton da Cunha Cavalcanti
CIn/UFPE

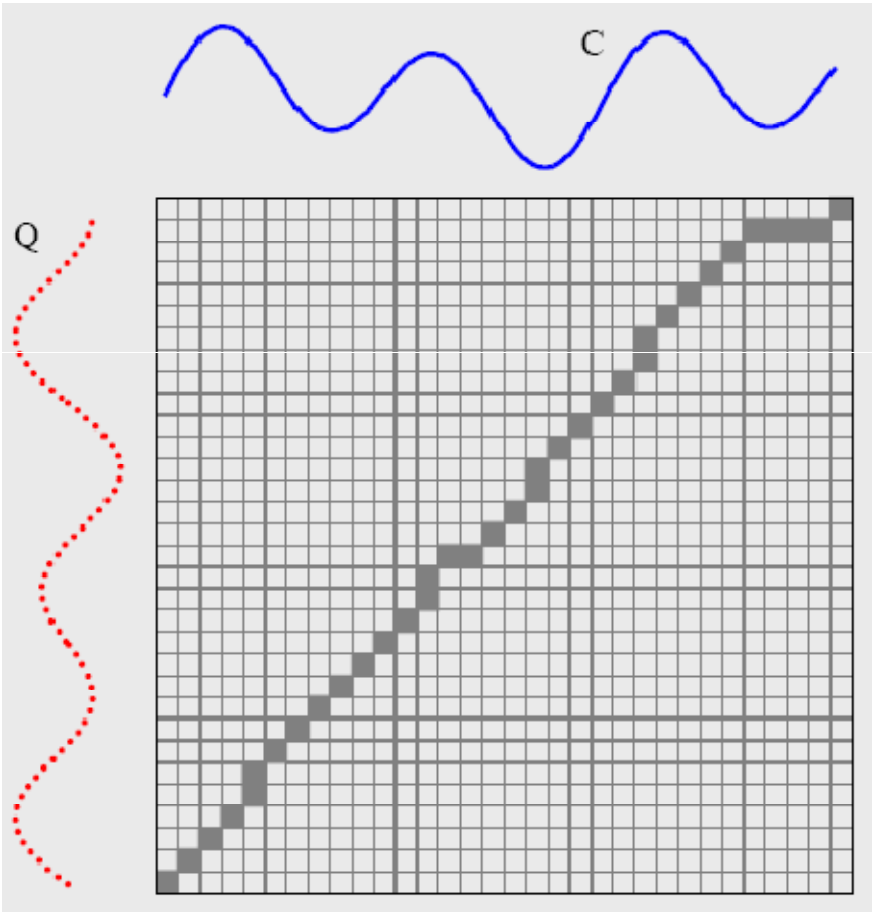
Introdução

- Objetivo: dado um conjunto de padrões de referência chamado de TEMPLATES, encontrar qual deles melhor se ajusta a um padrão desconhecido
  - Nesse caso, cada classe é representada por um padrão
- O ponto crucial é adotar uma medida apropriada para quantificar similaridade
- Essas medidas devem acomodar, de forma eficiente, variações entre os padrões de *template* e de teste
  - Por exemplo, a palavra "beauty" pode ter sido lida como "beeauty" ou "beuty", etc., devido a erros


Euclidian
DTW


1
0.8
0.6
0.4
0.2
0
C
Q

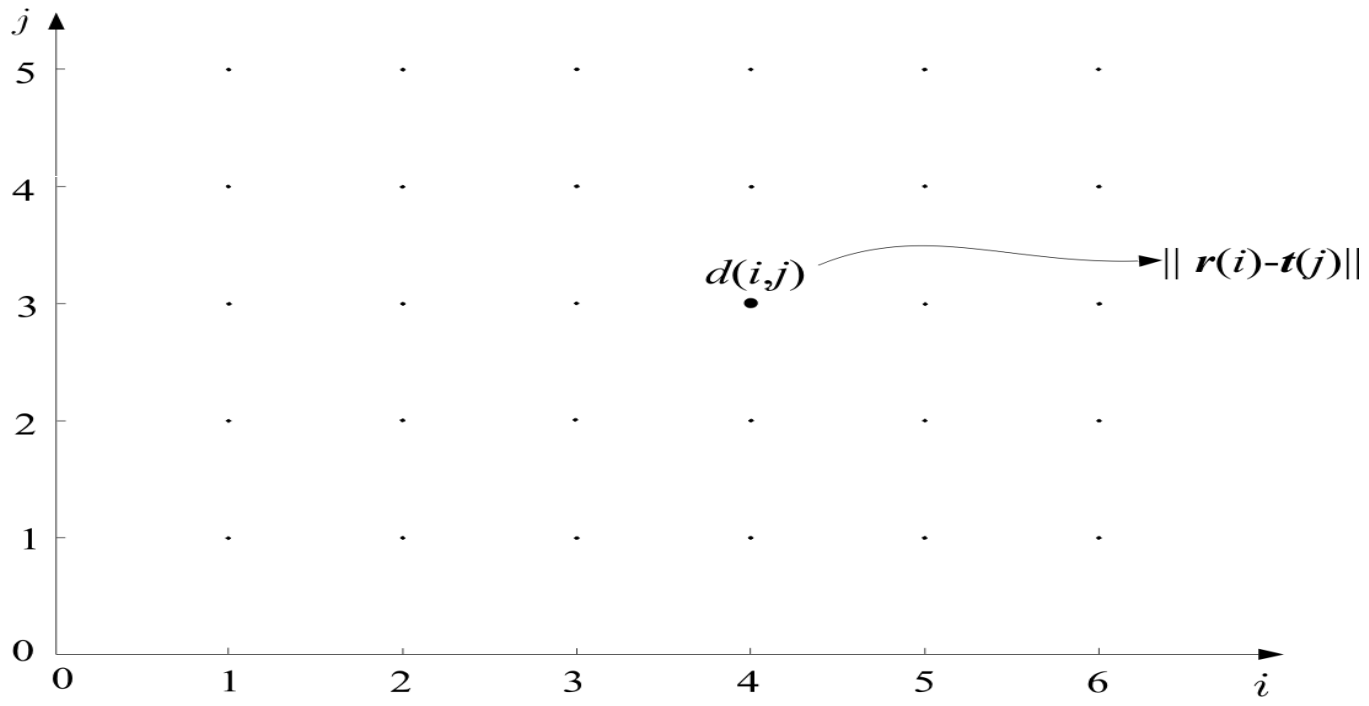

Q
C


C
Q

Introdução

- **Medidas baseadas em técnicas de busca do caminho ótimo**
  - Representação: o template é representado por uma seqüência de valores
  - Template: $\underline{r}(1), \underline{r}(2),..., \underline{r}(I)$
  - Test pattern: $\underline{t}(1), \underline{t}(2),..., \underline{t}(J)$

- Geralmente $I \neq J$
- Forma-se uma grade com I pontos (*template*) na horizontal e J pontos (teste) na vertical
- Cada ponto (i,j) da grade mede a distância entre $\underline{r}(i)$ e $\underline{t}(j)$

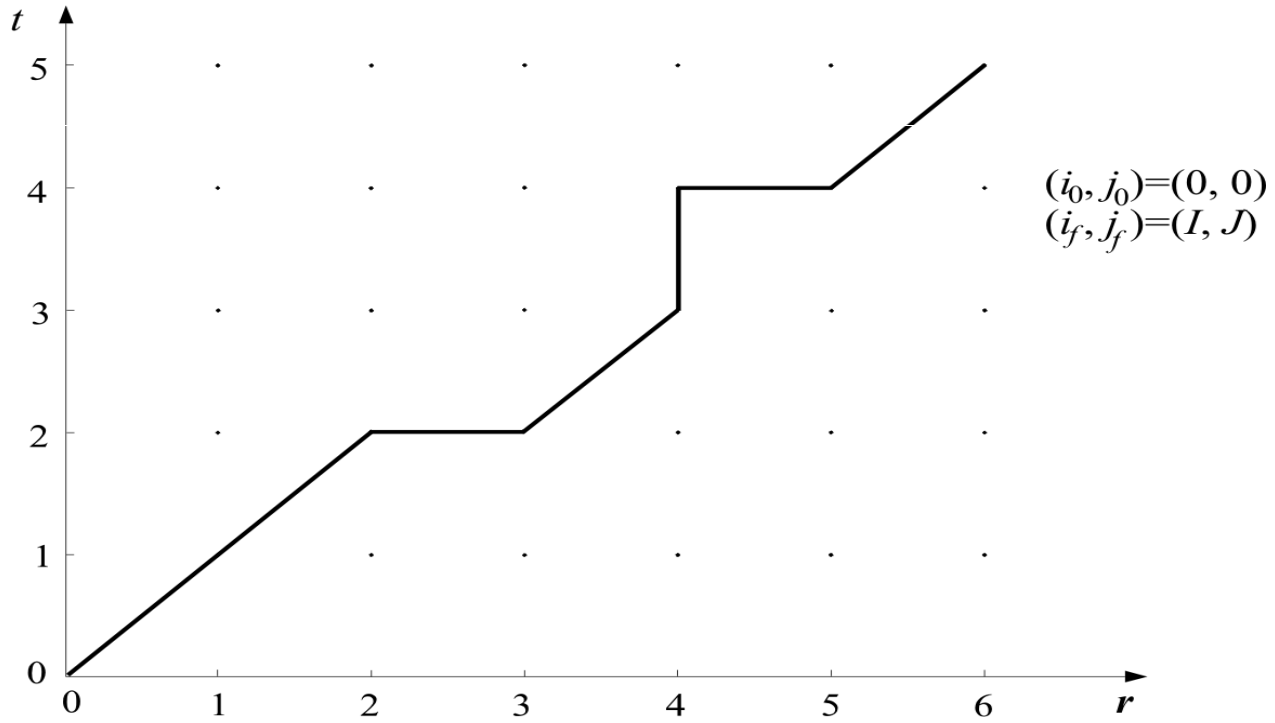


- Caminho: um caminho na grade, de um nó inicial (i_0, j_0) para um final (i_f, j_f), é um conjunto ordenado de nós $(i_0, j_0), (i_1, j_1), (i_2, j_2) \ldots (i_k, j_k) \ldots (i_f, j_f)$

- Cada caminho está associado a um custo

$$D = \sum_{k=0}^{K-1} d(i_k, j_k)$$

sabendo que K é o número de nós no caminho



- Buscar o caminho com custo ótimo $D_{opt.}$
- O custo entre os padrões de template $\underline{r}$ e de teste $\underline{t}$ é $D_{opt.}$

Bellman's Optimality Principle

- **Caminho ótimo:**

$$(i_0, j_0) \xrightarrow{opt} (i_f, j_f)$$

- **Seja *(i,j)* um nó intermediário**

$$(i_0, j_0) \rightarrow \ldots \rightarrow (i, j) \rightarrow \ldots \rightarrow (i_f, j_f)$$

Então, o caminho ótimo passa por (i, j)

$$(i_0, j_0) \xrightarrow[(i,j)]{opt} (i_f, j_f)$$

- **Bellman's Principle:**

$$(i_0, j_0) \xrightarrow{opt} (i_f, j_f) = (i_0, j_0) \xrightarrow{opt} (i, j) \oplus$$
$$(i, j) \xrightarrow{opt} (i_f, j_f)$$

- **Em palavras: o caminho ótimo completo de (i_0,j_0) para (i_f,j_f) que passa por (i,j) é a concatenação do caminho ótimo de (i_0,j_0) para (i,j) com o caminho de (i,j) para (i_f,j_f)**

- **Seja $D_{opt.}(i,j)$ o caminho ótimo para alcançar (i,j) de (i_0,j_0), assim o princípio de *Bellman* define que:**

$$D_{opt}(i_k, j_k) = opt\{D_{opt}(i_{k-1}, j_{k-1}) + d(i_k, j_k)\}$$

The Edit distance

- **É usada para *matching* entre palavras escritas.**
  Aplicações:
  - Edição Automática
  - Recuperação de Texto
- **A medida dele levar em consideração:**
  - Troca de símbolos, “befuty” ao invés de “beauty”
  - Erros de inserção, “bearuty”
  - Erros de remoção, “beuty”

- **O custo é baseado na conversão de um padrão em outro**

- ***Edit distance*: Número total mínimo de mudanças, *C*, inserções *I* e remoções *R*, necessárias para transformar o padrão *A* no padrão *B*,**

$$D(A,B) = \min_{j}[C(j) + I(j) + R(j)]$$

sabendo que *j* varre todas as possíveis variação de símbolos, objetivando converter *A* *B*

- **Cálculo do custo**

  - $(i-1, j-1) \rightarrow (i, j)$

$$d(i, j \mid i-1, j-1) = \begin{cases} 0, & if\ t(i) = r(j) \\ 1, & t(i) \neq r(j) \end{cases}$$

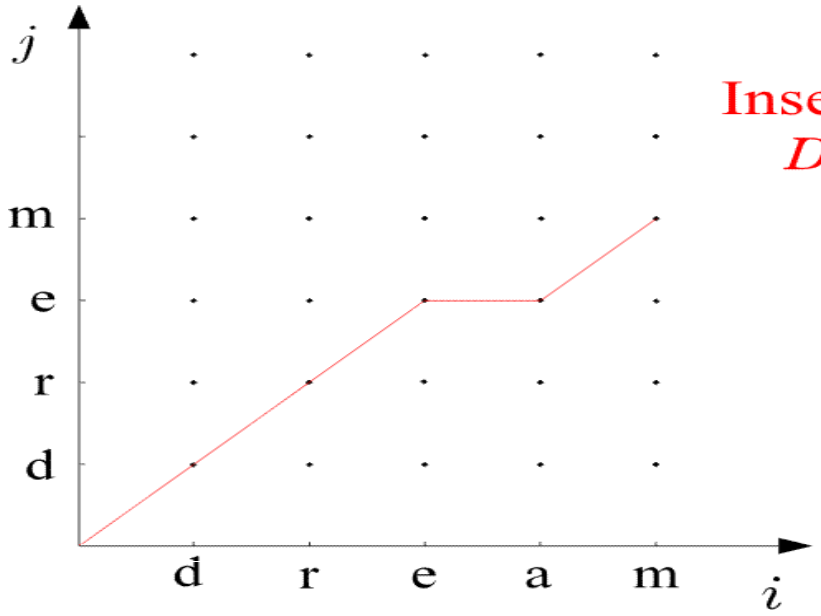
  - Horizontal

$$d(i, j \mid i-1, j) = 1$$

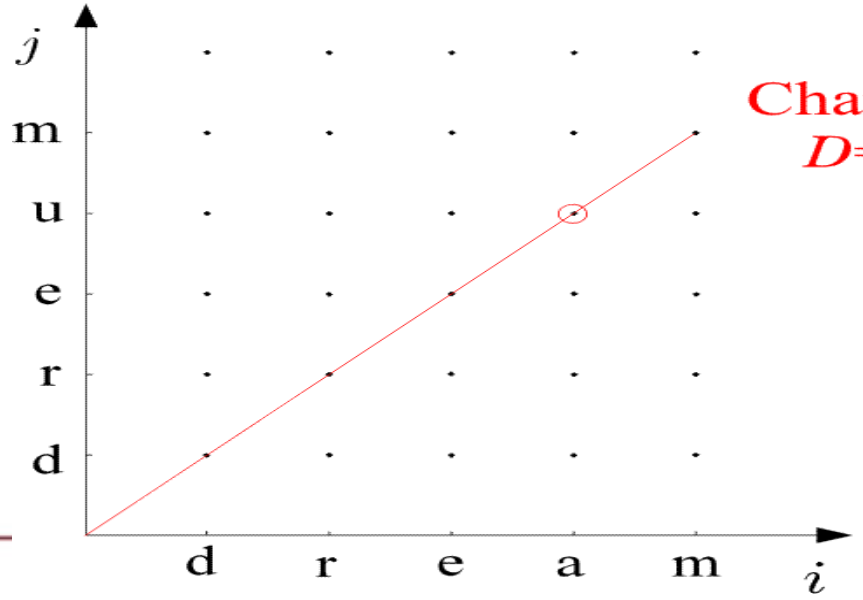
  - Vertical

$$d(i, j \mid i, j-1) = 1$$

Exemplos



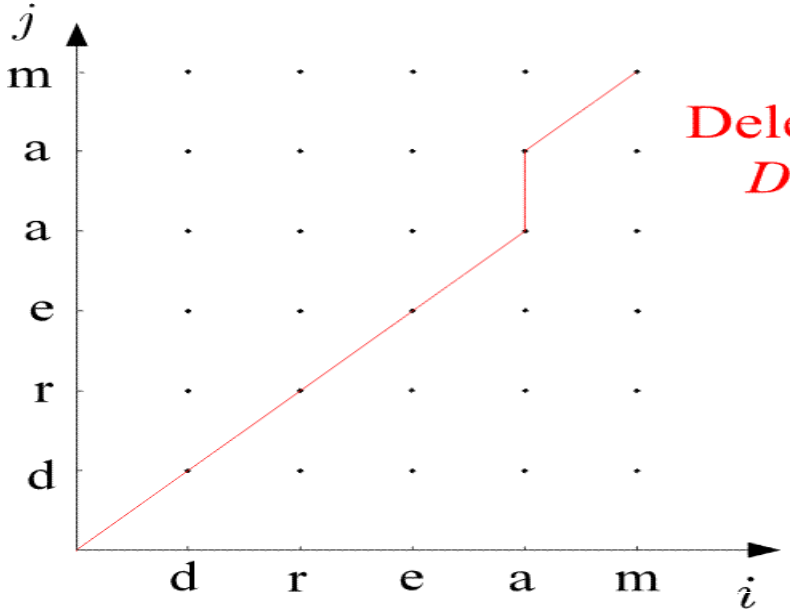
Insertion
$D=1$

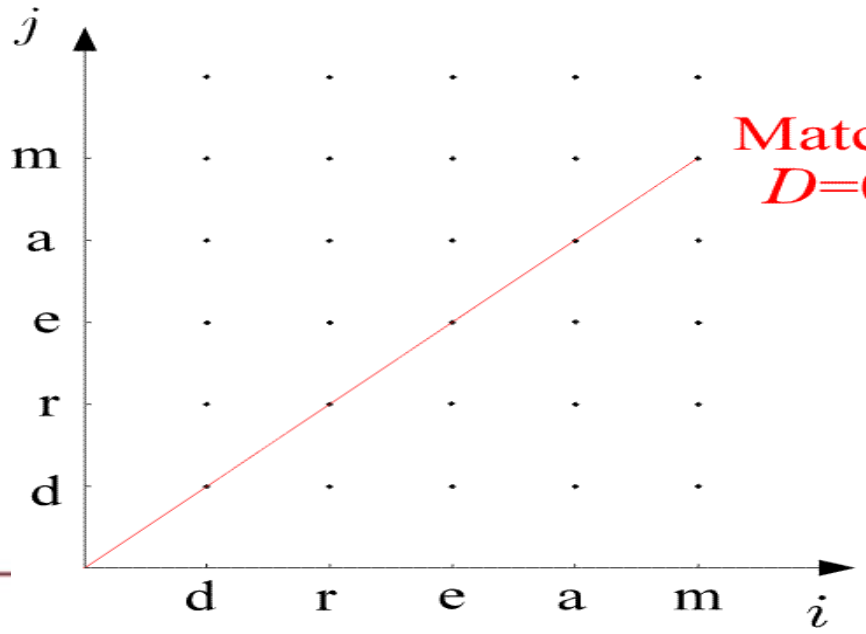


Change
$D=1$

Centro de Informática

Exemplos





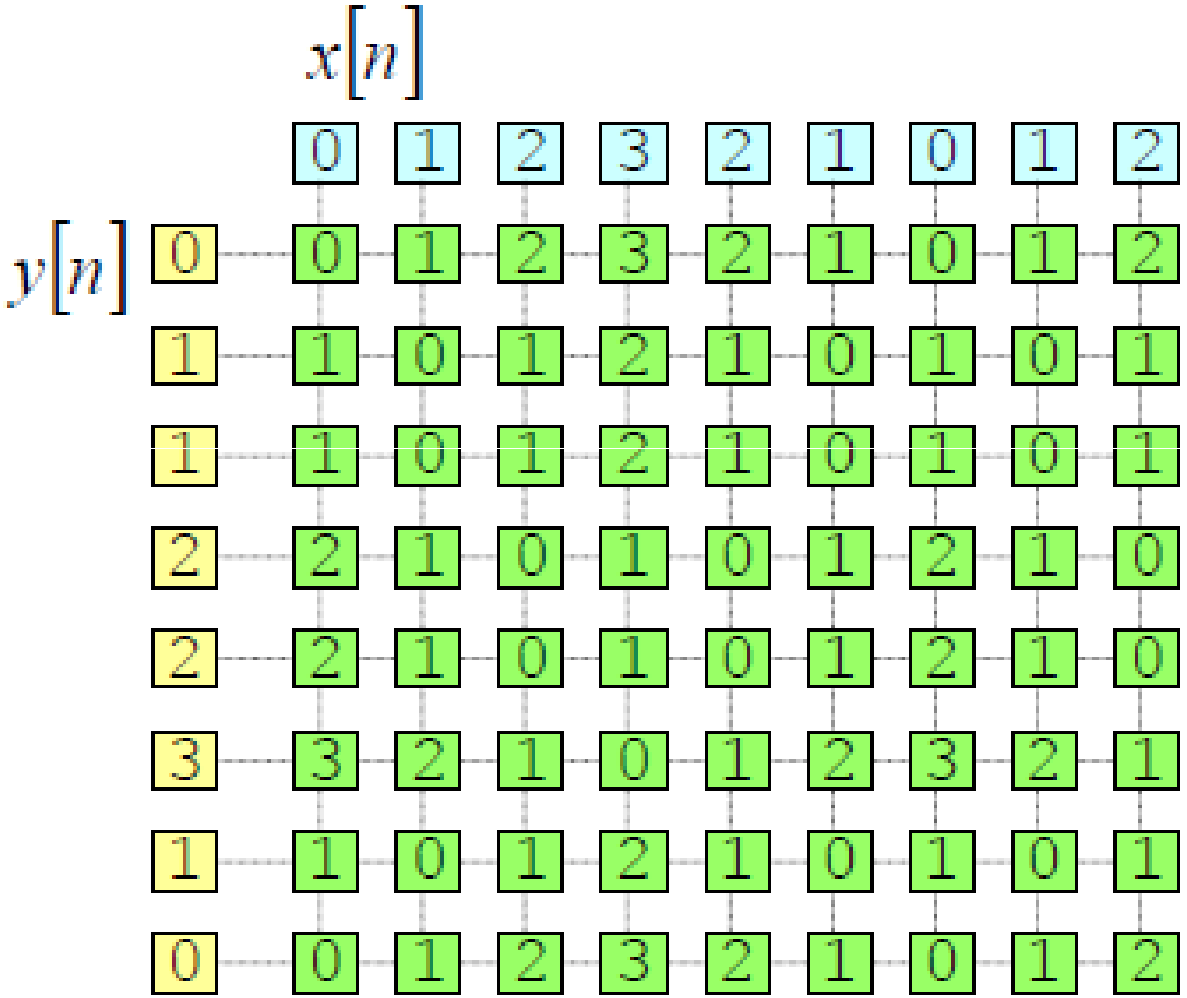
O algoritmo

- $D(0,0)=0$
- $For\ i=1,\ to\ I$
  - $D(i,0)=D(i-1,0)+1$
- $END\ \{FOR\}$
- $For\ j=1\ to\ J$
  - $D(0,j)=D(0,j-1)+1$
- $END\{FOR\}$
- $For\ i=1\ to\ I$
  - $For\ j=1,\ to\ J$
    - $C_1=D(i-1,j-1)+d(i,j \mid i-1,j-1)$
    - $C_2=D(i-1,j)+1$
    - $C_3=D(i,j-1)+1$
    - $D(i,j)=min\ (C_1,C_2,C_3)$
  - $END\ \{FOR\}$
- $END\ \{FOR\}$
- $D(A,B)=D(I,J)$

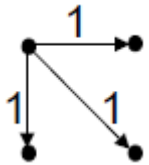
Centro de Informática U·F·P·E

Exemplo

$$d(i,j) = |x[i] - y[j]|$$



Exemplo



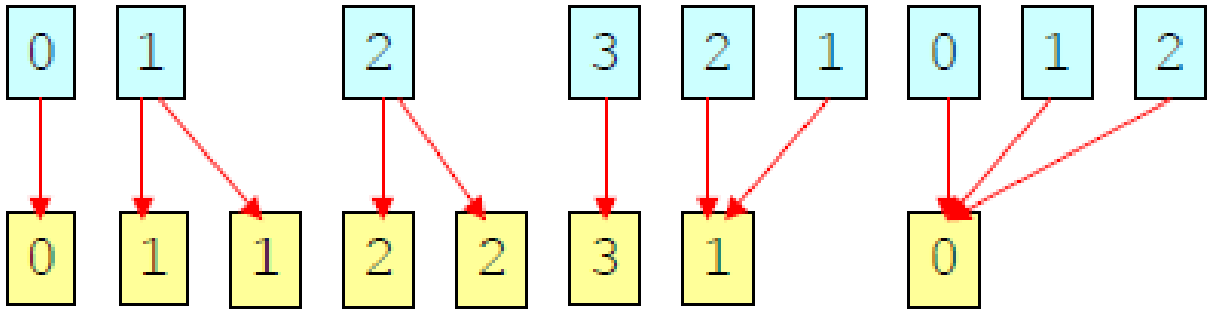
$x[n]$

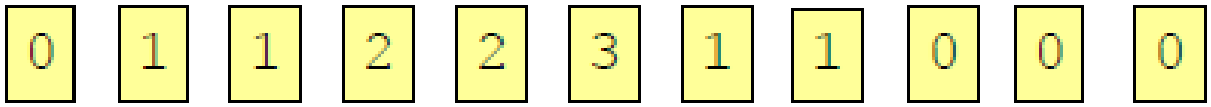
$y[n]$	0	1	2	3	2	1	0	1	2
0	0	1	3	6	8	9	9	10	12
1	1	0	1	3	4	4	5	5	6
1	2	0	1	3	4	4	5	5	6
2	4	1	0	1	1	2	4	5	5
2	6	2	0	1	1	2	4	5	5
3	9	4	1	0	1	3	5	6	6
1	10	4	2	2	1	1	2	2	3
0	10	5	4	5	3	2	1	2	4

Exemplo

Centro de Informática UFPE

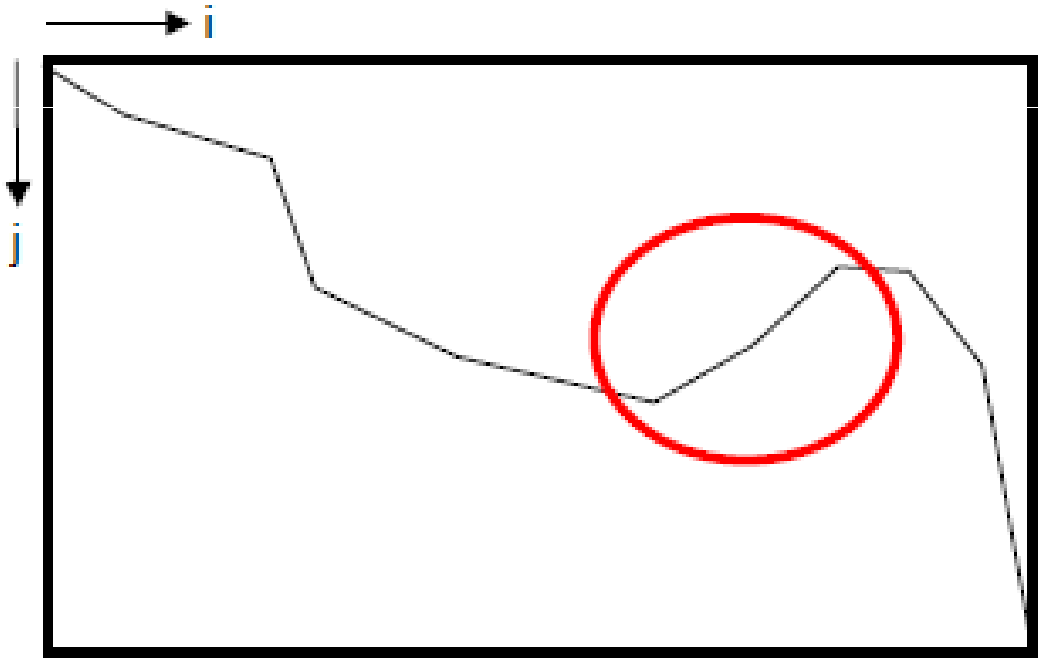
Exemplo





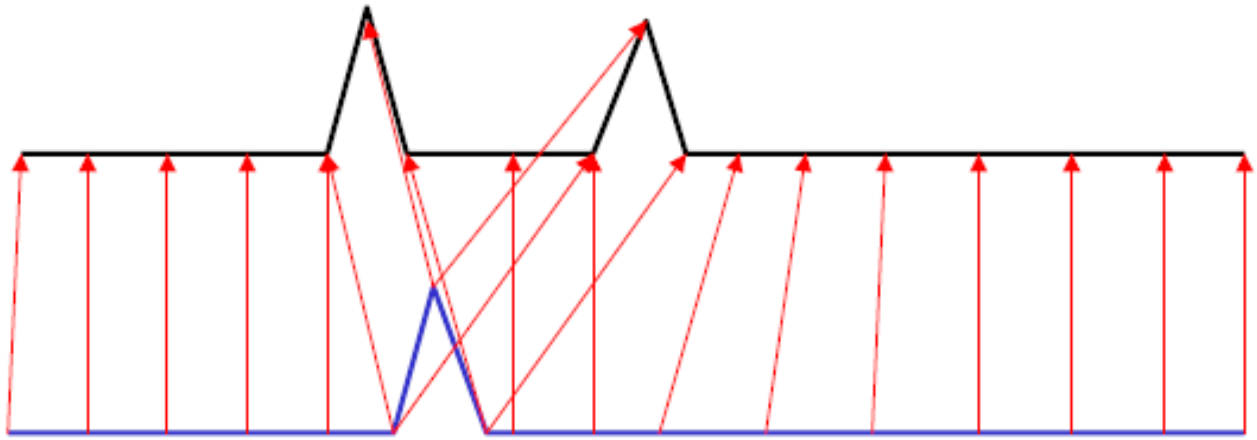
Propriedades

- **Monotonicidade**
  - O caminho percorrido pelo alinhamento não pode voltar no “tempo”



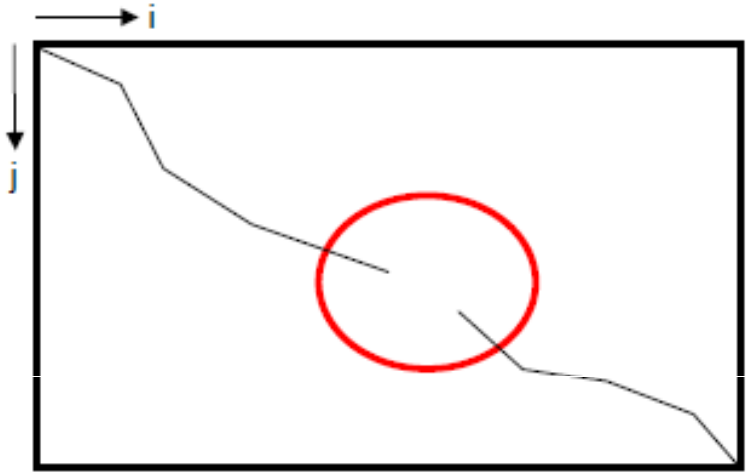
Propriedades

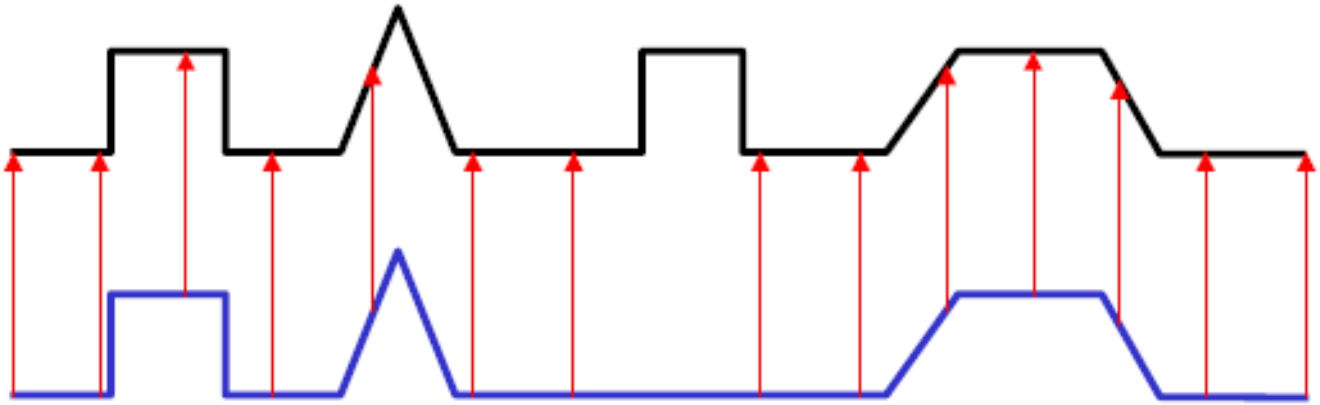
- **Pois, se o alinhamento não for monotônico algumas características podem ser repetidas**
- **Assim, a função de dissimilaridade não teria sentido**



Propriedades

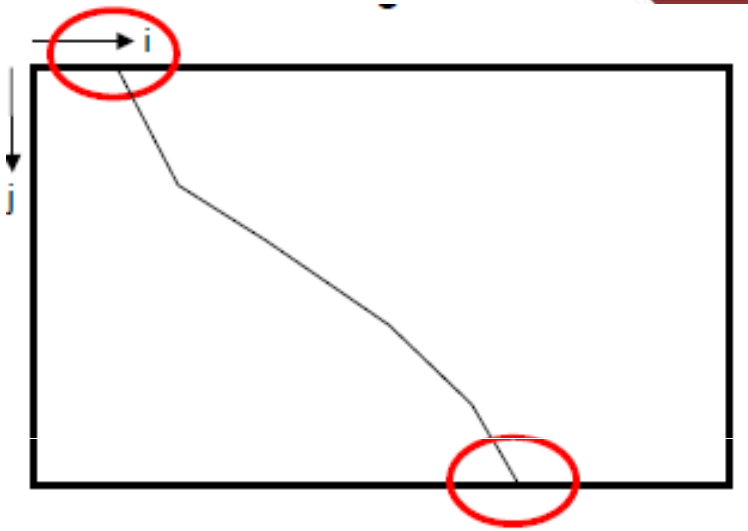
- **Continuidade**
  - Caso contrário o alinhamento irá desprezar características importantes

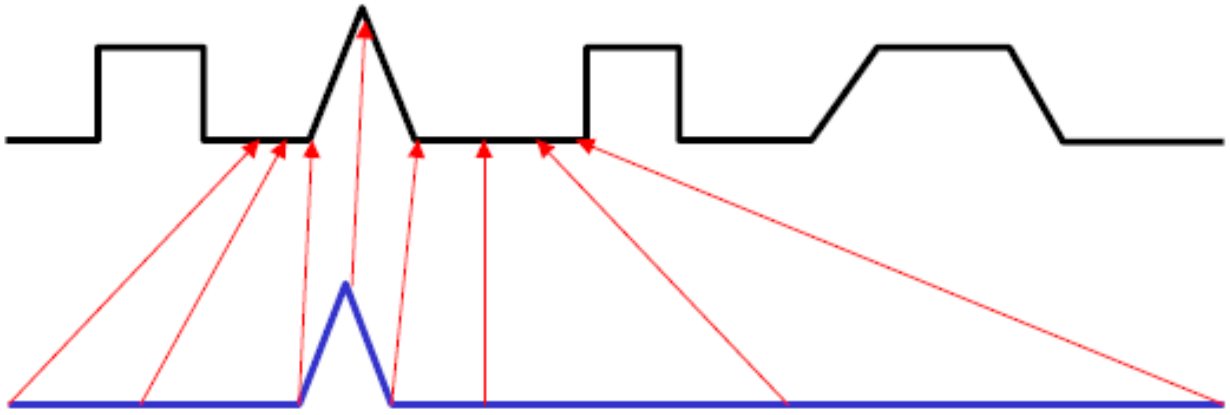




Propriedades

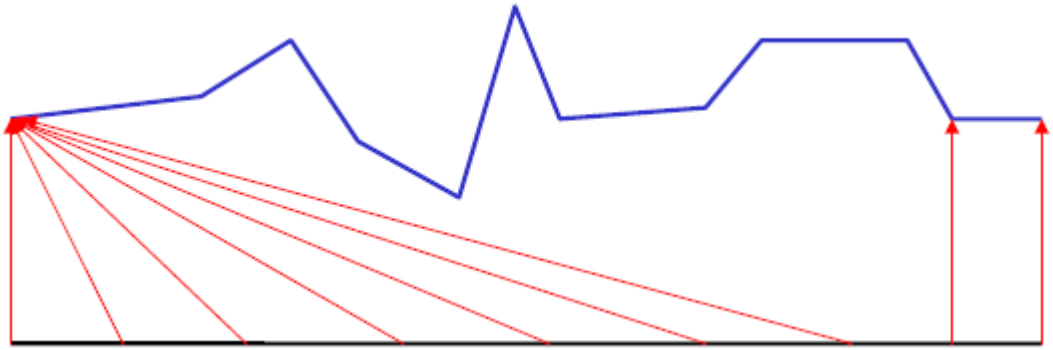
- **Limites**
  - O alinhamento deve começar e terminar em pontos específicos





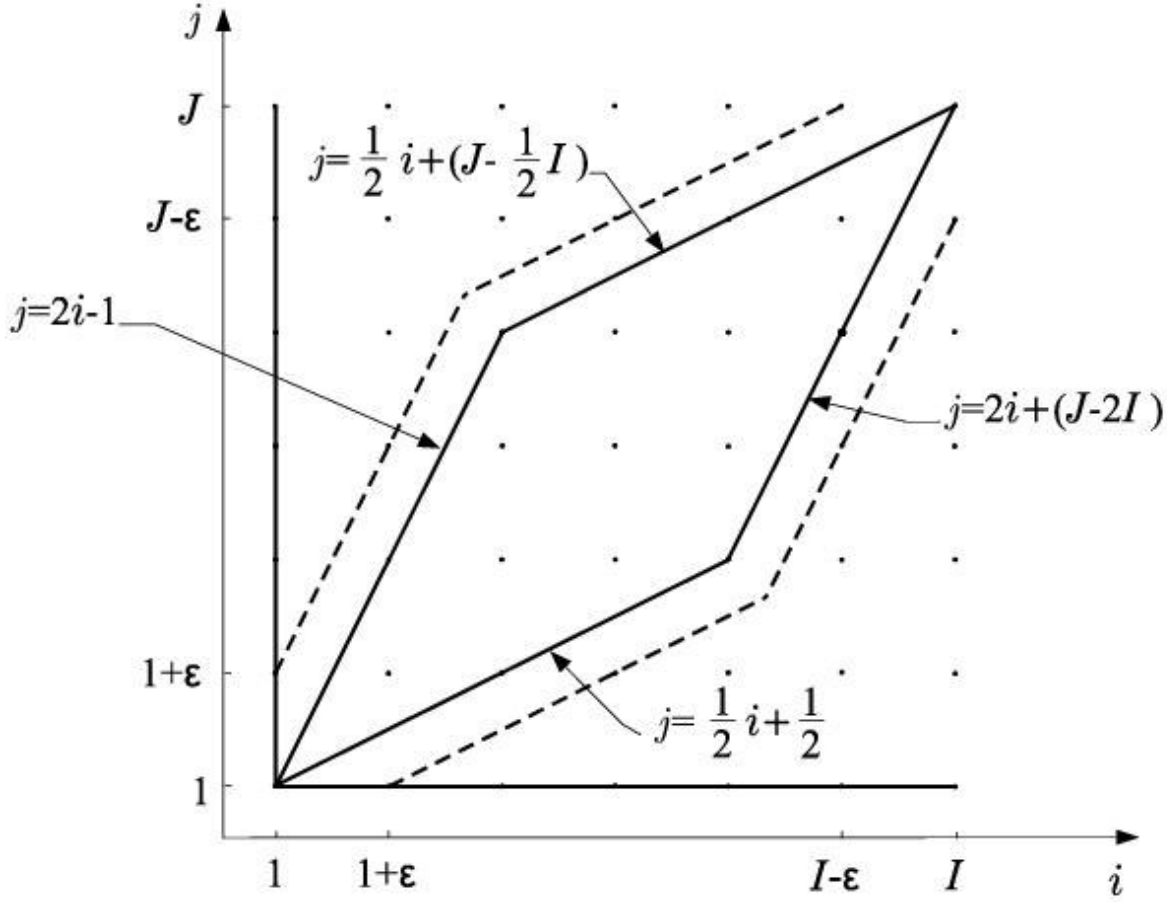
Propriedades

- **Janela de busca**
  - O alinhamento deve estar relativamente perto do caminho “identidade”

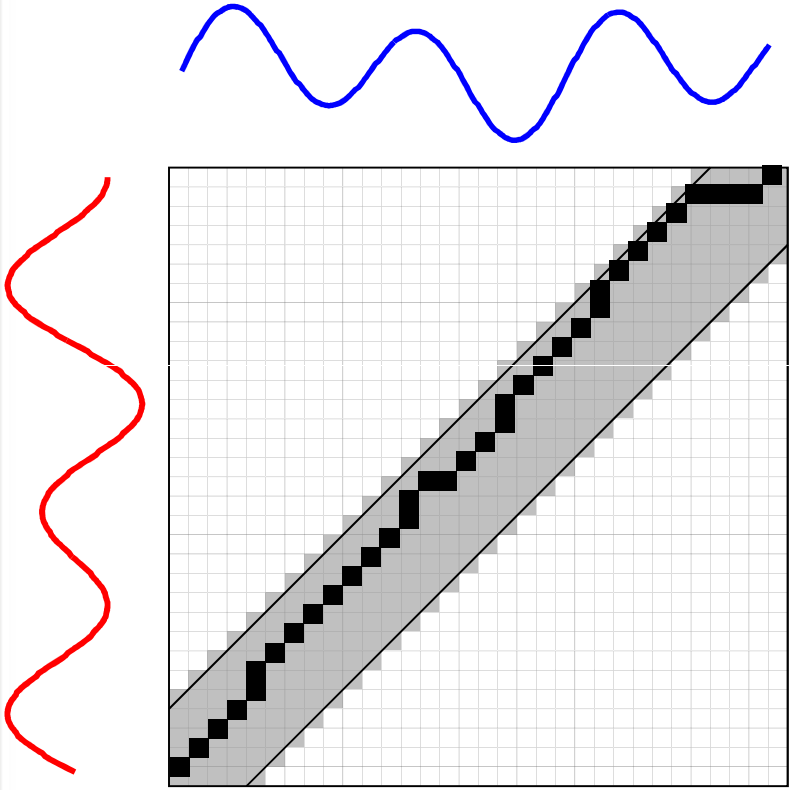


Restrição Global

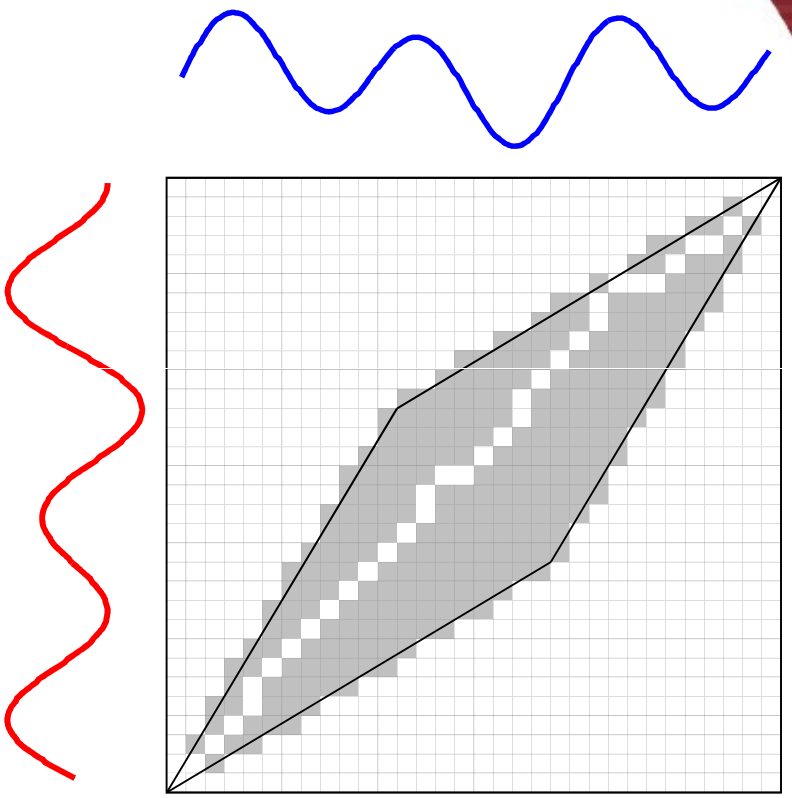
- **Define uma região do espaço a partir da qual a busca do caminho ótimo será realizada**



Restrição Global



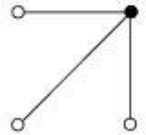
Sakoe-Chiba Band



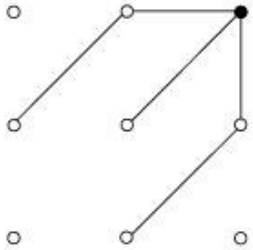
Itakura Parallelogram

Restrições Locais

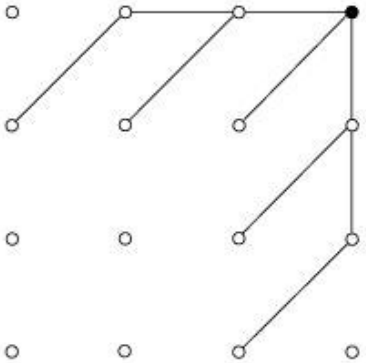
- **Define o tipo de transição permitida**



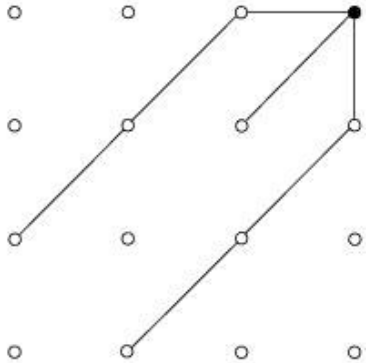
(a)



(b)

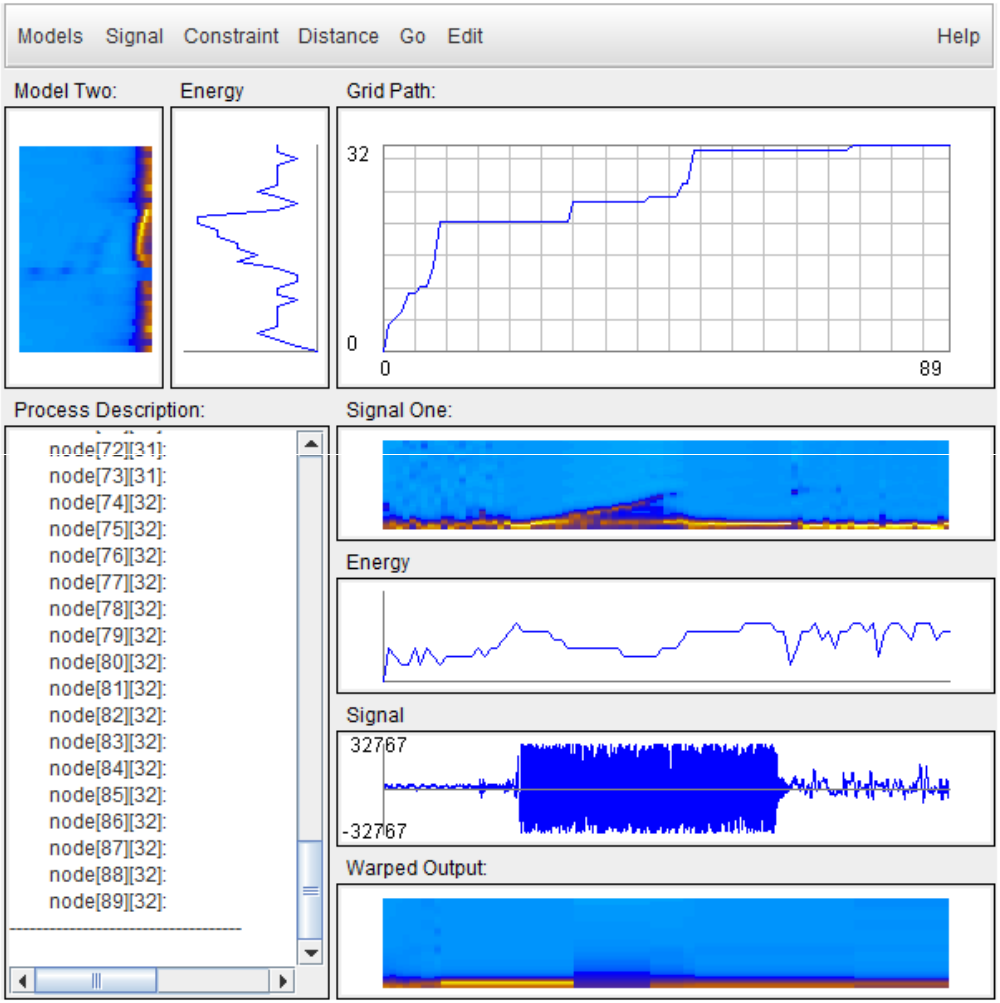


(c)



(d)

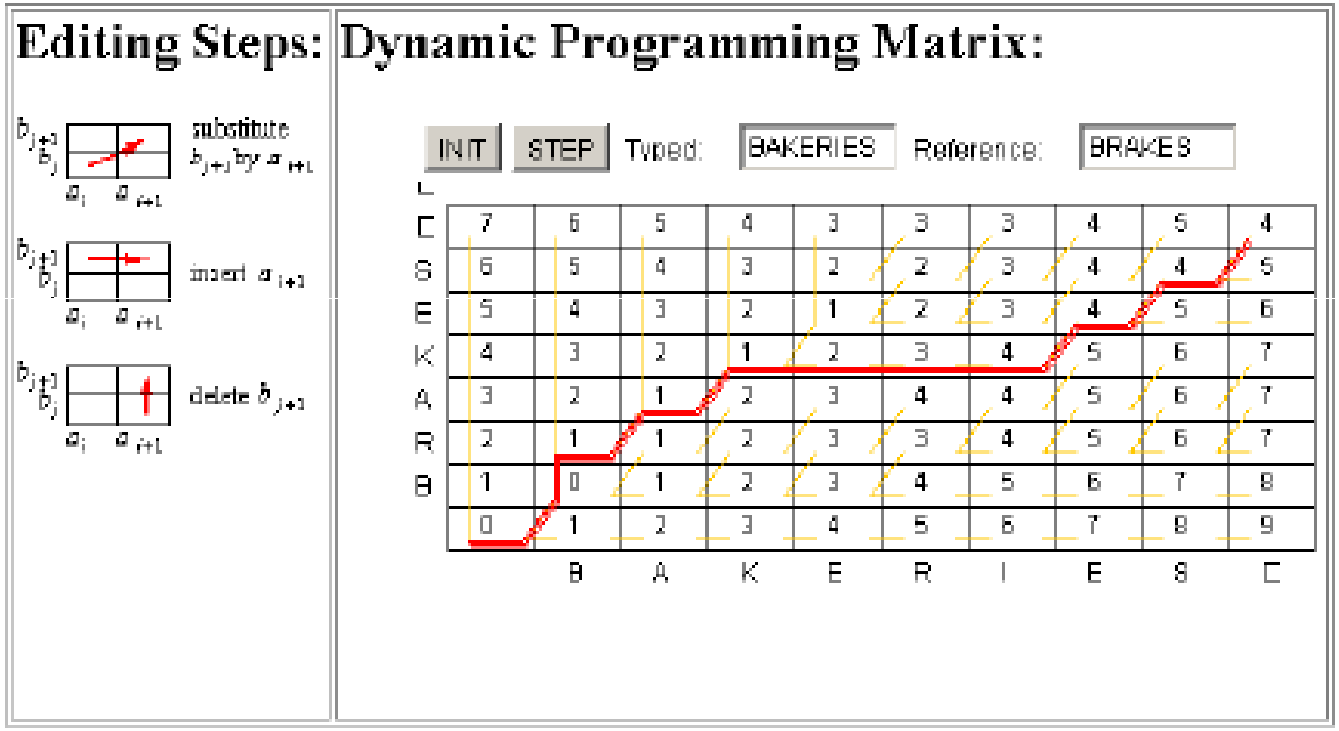
Aplicações – Reconhecimento de Fala



http://www.isip.piconepress.com/projects/speech/software/demonstrations/applets/

Centro de Informática U·F·P·E

Aplicação
String Matching – Alinhamento de cadeias de DNA



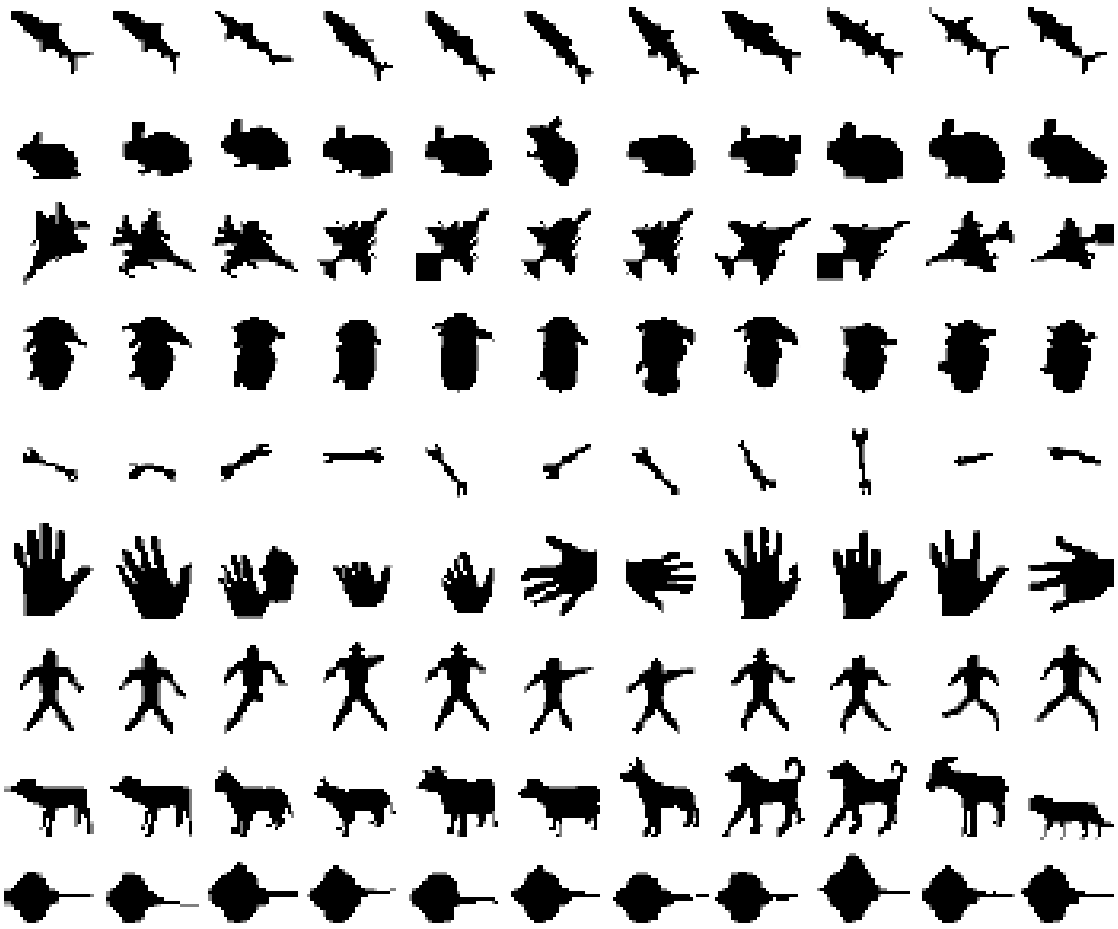
Aplicação – Reconhecimento de Caracteres Manuscritos

	 4	 7	 10	 13	 14
	 2	 4	 5	 6	 10
	 4	 11	11	 13	 15
	 2	 10	 13	 14	 15
	2	 8	 10	 11	 13
	 12	 17	 18	 20	 21
	 6	 8	 9	11	15
	5	12	19	19	19

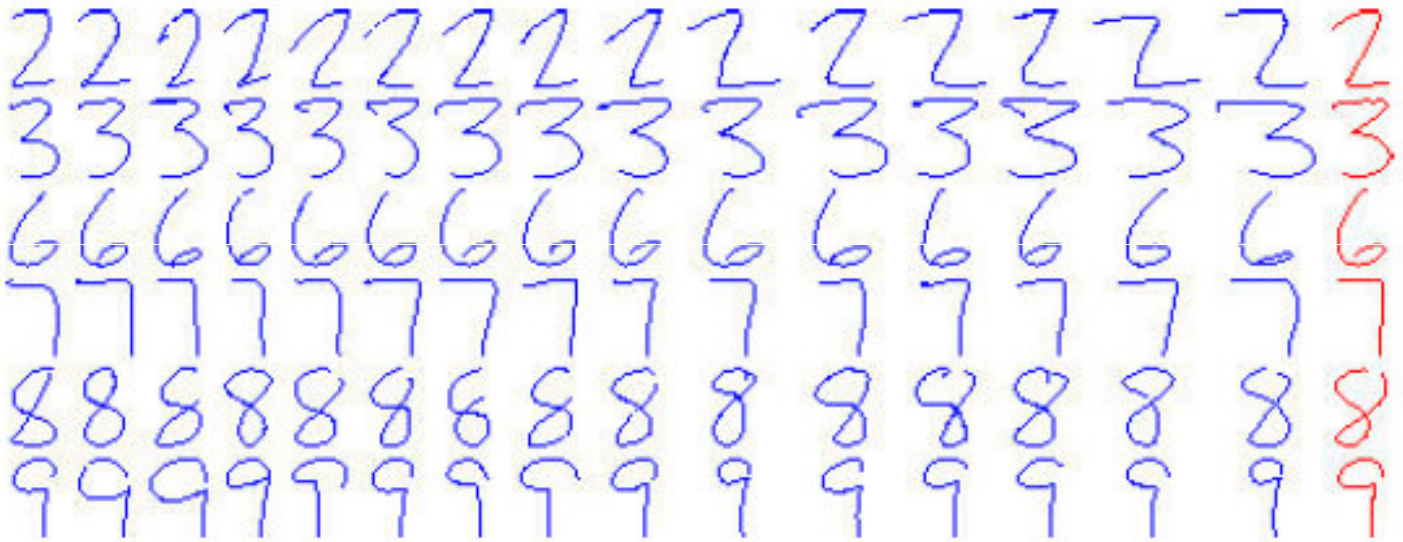
	4	11	15	16	18
	5	15	17	21	22
	4	15	16	18	18
	4	17	18	18	19
	4	10	11	11	12
	5	8	11	12	13
	6	16	17	18	21
	3	7	10	13	14

Centro de Informática

Aplicação – Reconhecimento de Objetos

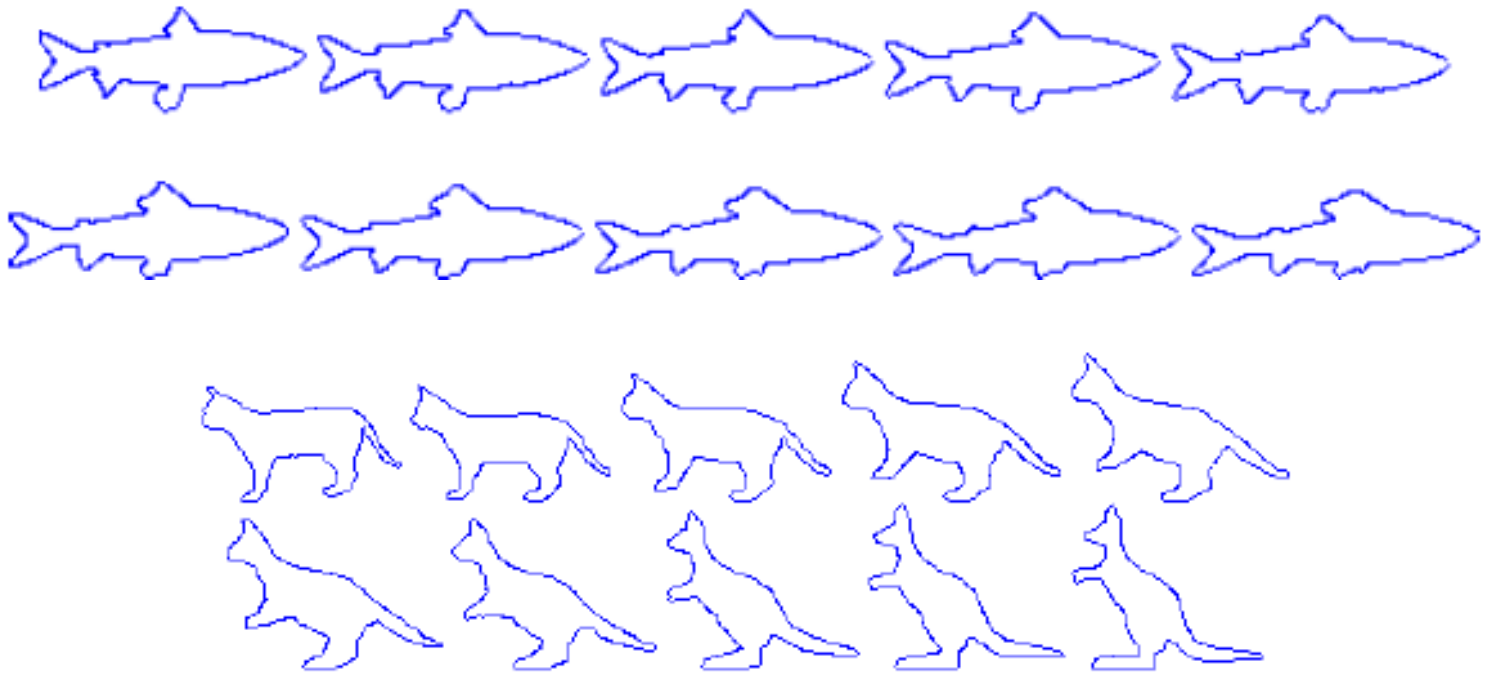


Aplicação – Geração de Protótipos

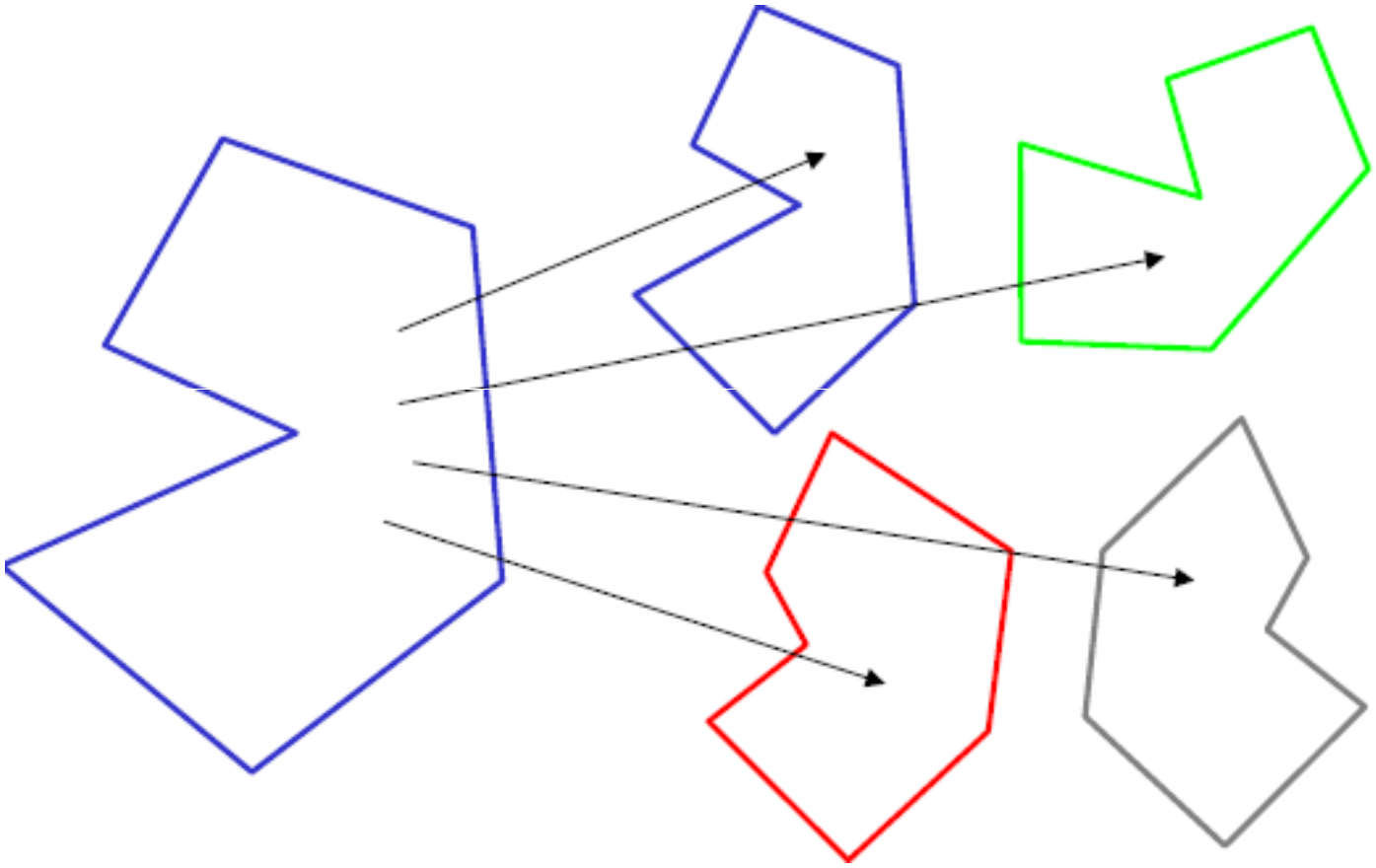


Centro de Informática U·F·P·E

Aplicação – Morphing



Aplicação – Reconhecimento de Polígonos



Considerações Finais

- Dymanic Time Warping (DTW) é uma técnica robusta para medir similaridade entre séries temporais
- DTW é muito usada em várias áreas
- DTW é bastante custoso, em termos de tempo computacional. Mas abordagens para aumentar a velocidade foram desenvolvidas: restrições globais

Referências

- S. Theodoridis and K. Koutroumbas. **Pattern Recognition**. Academic Press. 2006.
- E. Keogh and C. A. Ratanamahatana. *Exact indexing of dynamic time warping*. **Journal of Knowledge and Information Systems**. 7(3), pp.358-386, 2005.
- J. Aach and G. Church. *Aligning gene expression time series with time warping algorithms*. **Bioinformatics** 17:495–508, 2001.
- H. Sakoe and S. Chiba. *Dynamic programming algorithm optimization for spoken word recognition*. **IEEE Transactions on Acoustics, Speech, and Signal Processing**. 26(1), pp.43-49, 1978.
- L. Rabiner, A. Rosenberg, S. Levinson. *Considerations in dynamic time warping algorithms for discrete word recognition*. **IEEE Transactions on Acoustics, Speech, and Signal Processing**, 26:575–582, 1978.
- C. Myers, L. Rabiner, A. Roseneberg. *Performance tradeoffs in dynamic time warping algorithms for isolated word recognition*. **IEEE Transactions on Acoustics, Speech, and Signal Processing**, 28:623–635, 1980.