Algoritmusok és Adatszerkezetek II.
utolsó előadás
Beszédtechnológiai algoritmusok

(csak egy kis felszínkapargatás)

Beszédtechnológia

- Eredeti feladat: beszédfelismerés
  - Input: beszédjel (mikrofonon át)
  - Output: szöveges leirat (mit mondott?)
  - Lehetőleg minél pontosabb…
- Ma már tágabb a beszédtechnológia
  - Pl. konfliktusfelismerés, orvosi diagnosztika (Alzheimer, Parkinson, EKZ)
  - Egyéb paralingvisztikus („nyelven túli”) információk kinyerése, pl. fizikai terhelés mértéke, anyanyelv meghatározása (akcentusfelismerés), őszinteség meghatározása (hazugságfelismerés), fogyasztott étel…

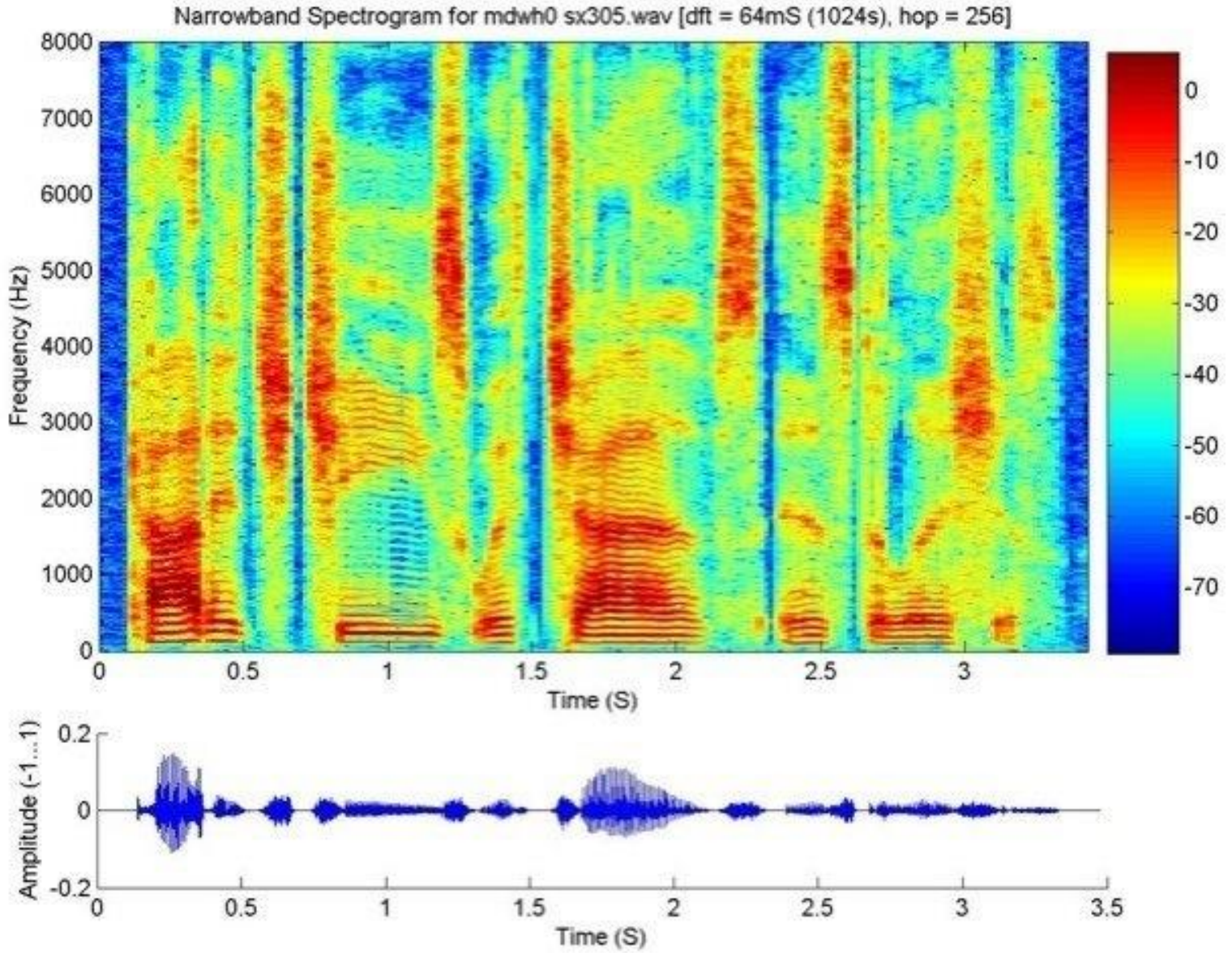
Dinamikus idővetemítés (DTW)

- Dynamic Time Warping (DTW): felismerési megközelítés az ősidőkből
  - (H. Sakoe & S. Chiba (1978): "Dynamic programming algorithm optimization for spoken word recognition." *Transactions on ASSP* **26**: 43–49.)
- Izolált szavas megközelítés: nem szósorozatot akarunk felismerni, csak egy-egy szót (pl. parancsok)
- Minden felismerhető szóhoz eltárolunk egy példát
- A bemondott szót minden felismerhető szóval összehasonlítjuk (nem gyors, de kevés szónál akár jó is lehet…)

Hogyan vetjük össze a két szót?

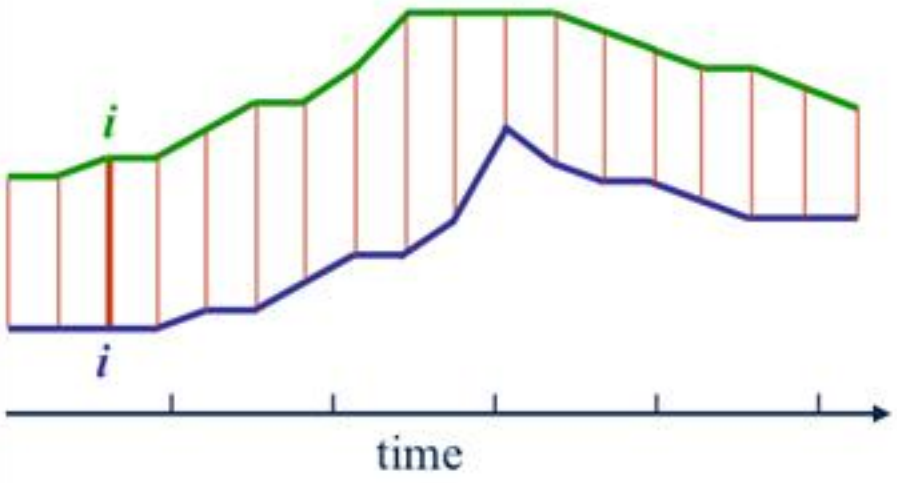
- Problémák:
  - eltérő hosszú szavak
  - lokális hasonlóság?
- Lokális összevetés:
  - Spektrális felbontás (Fast Fourier Transformation, FFT)
    - Adott ponton adott (audio)frekvencia mennyire erős
  - Rövid (pl. 10ms), egyenlő hosszú darabokra vágjuk (*frame*)
    - Ez durva (pontatlan) megközelítés, de most elég nekünk ez is…
  - Frame-szinten hasonlítunk
    - c(i,j): egyik felvétel i. frame-e mennyire hasonlít a másik felvétel j. frame-ére? ($c(i,j) \geq 0$; $c(i,j)=0 \leftrightarrow$ a két frame megegyezik)
    - Gépi tanulás (pl. GMM, ANN): majd Mest.Int-ből
    - Négyzetes eltérés, euklideszi távolság, Manhattan-távolság…

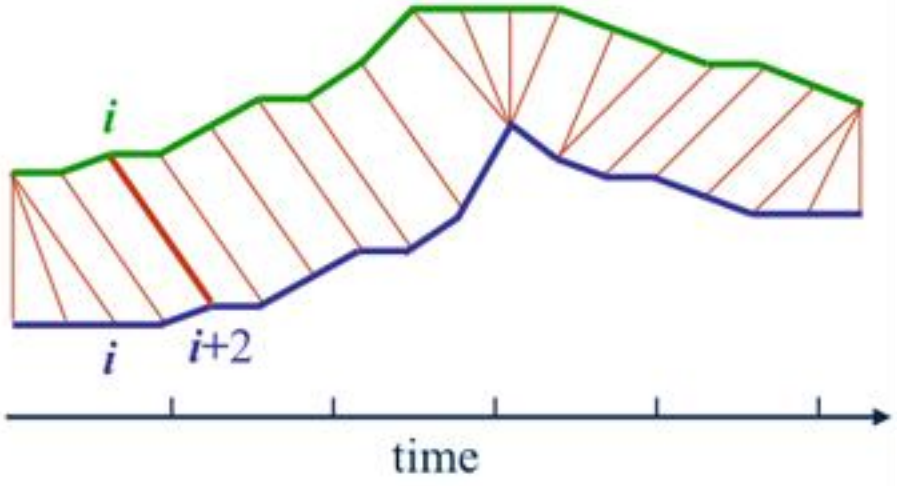
Spektrális felbontás



Hogyan vetjük össze a két szót?

- Tehát meg tudjuk mondani, hogy két frame (akár két különböző szóból is) mennyire hasonló
- De még mindig eltérő hosszú szavakat hasonlítunk!
  - Frame-ben mérve is
- Ugyanazt a szót ugyanaz a beszélő egymás után is különbözőképpen ejti
  - A rövidülés/hosszabbítás nem egyenletes
  - Pl. egyes hangzókat megnyújtunk, hangsúlyozunk stb.



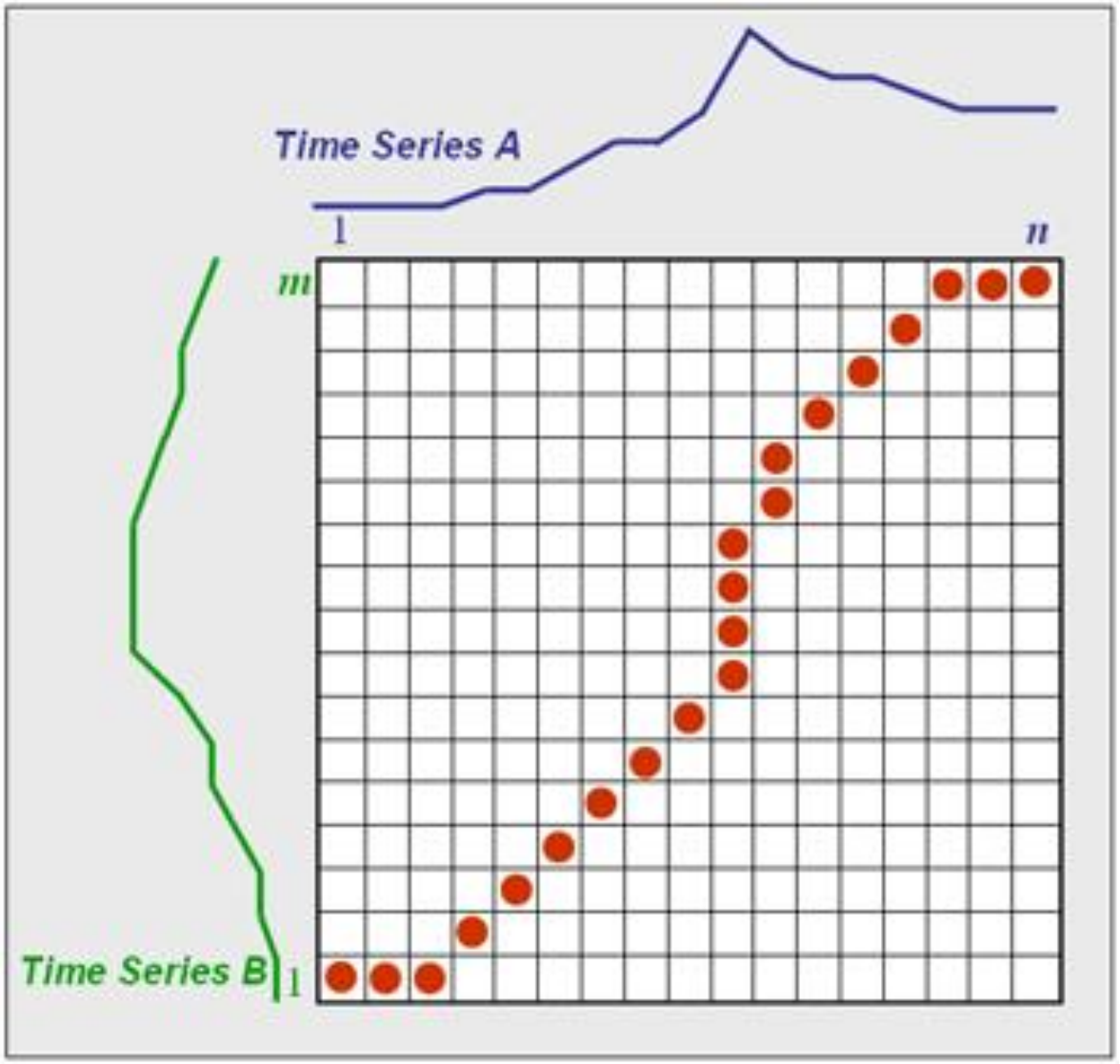


Any distance (Euclidean, Manhattan, …) which aligns the *i*-th point on one time series with the *i*-th point on the other will produce a poor similarity score.

A non-linear (elastic) alignment produces a more intuitive similarity measure, allowing similar shapes to match even if they are out of phase in the time axis.

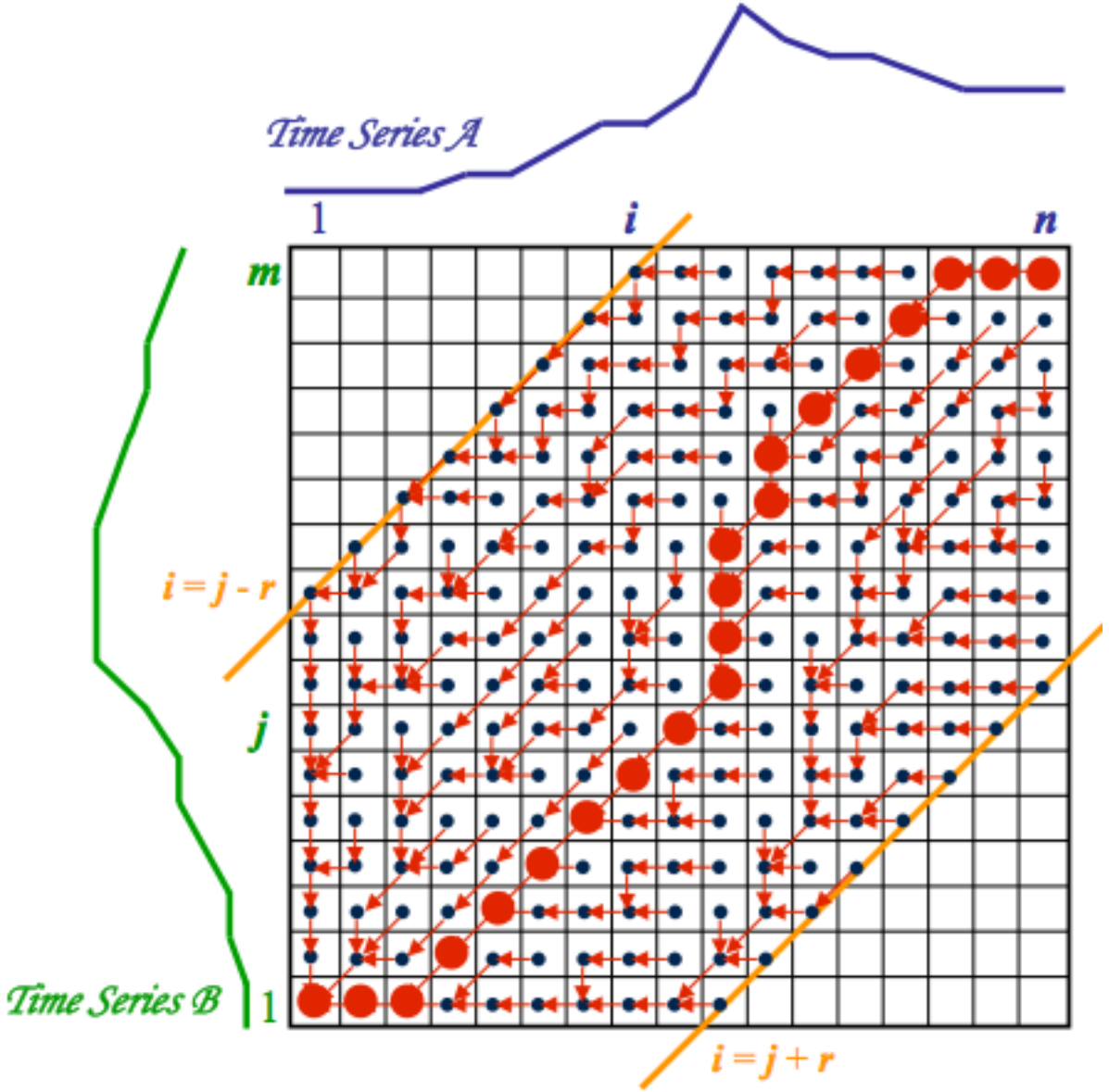
DTW (még mindig)

- DTW ötlete: az egyes részeket időben („time”) megnyújtjuk („warping”)
  - Sőt, mint majd látjuk, dinamikusan is…
- A kiejtett szó *i*. frame-jét az etalon felvétel *j*. frame-jével hasonlítjuk össze
- Nem össze-vissza: mindkét szóban csak előrefelé léphetünk vagy megállhatunk
- Frame-szintű távolság (*c(i,j)*) helyett az egész út **összköltsége** érdekes
  - Minimális összköltségű utat keresünk
  - Dinamikus algoritmus! Ld. 1. félév


Time Series A
1
n
m
Time Series B
1

Feltételek

- **Monotonitás**: az i és j indexek sosem csökkennek, csak nőnek vagy változatlanok maradnak
- **Folytonosság**: nem ugrunk át frame-et; tehát i és j max. 1-gyel nőhet
- **Korlátosság** (*boundary*): az utak az 1. frame-eknél kezdődnek és az utolsó frame-eknél fejeződnek be
- **Ablakozás** (*window*): egy jó útvonal az átló környékén marad (max. *r* távolságra) (az algoritmus gyorsítása)
- **Emelkedés** (*slope*): egy jó útvonal ne legyen túl meredek vagy lapos (*k* db. egyik irányba tett lépés után lépjünk a másik irányba is)


Time Series A
1
i
n
m
i = j - r
j
Time Series B
1
i = j + r

Algoritmus

- d(i,j): optimális út összhossza a felvételek elejétől az i., illetve j. frame-ig
- $d(i,j) \geq 0,\ 1 \leq i \leq n,\ 1 \leq j \leq m$
- $d(1,j) = \Sigma c(1,k)$ for all $1 \leq k \leq j$
- $d(i,1) = \Sigma c(k,1)$ for all $1 \leq k \leq i$
- $d(i, j) = c(i,j) + \min(d(i, j-1), d(i-1, j), d(i-1,j-1))$
- d(n, m) az optimális út összköltsége
- Ebből optimális út: visszakeresés
  - De ez most nem kell

A legjobb szó kiválasztása

- Minden felismerhető szóra megkaptuk a legrövidebb út **össz**hosszát
- Ez alapján a szavak közül még nem tudunk választani!
  - Hosszú szóra vett távolság várhatóan (sokkal) több lesz
- Megoldás: normalizálás
  - Az utat elosztjuk a szó hosszával
  - Szimmetrikus: a két szó hosszának összegével
  - Aszimmetrikus: csak a bemondott szóéval

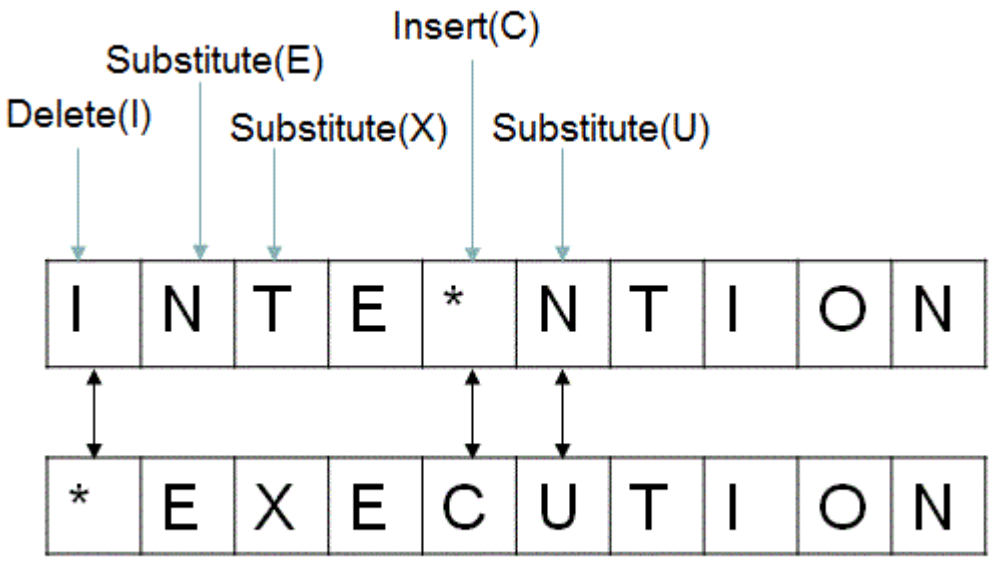
Illesztési távolság

- Illesztési távolság (*edit distance* vagy *Levenshtein distance*)
- Nem épp egy mai algoritmus
  - *Влади́мир И. Левенштейн (1965). Двоичные коды с исправлением выпадений, вставок и замещений символов. Доклады Академий Наук СССР **163** (4): 845–8.*
  - *Levenshtein, Vladimir I. (February 1966). "Binary codes capable of correcting deletions, insertions, and reversals". Soviet Physics Doklady **10** (8): 707–710.*

Alapprobléma

- Adott két string (szósorozat, proteinszekvencia, stb.)
  - Tokenek (karakterek, proteinek, szavak stb.: véges! halmaz elemei)
  - Nem feltétlenül azonos hosszúak!
- Mennyire hasonló a kettő egymáshoz?
- Pontosabban: milyen átalakításokkal kaphatjuk meg egyiket a másikból, ahol a megengedett lépések:
  - Egy token (karakter, protein, szó) beszúrása
  - Egy token törlése
  - Egy token kicserélése egy másikra

Példa


Insert(C)
Substitute(E)
Delete(I)
Substitute(X)
Substitute(U)
I N T E * N T I O N
* E X E C U T I O N

Algoritmus

- Ez is standard dinamikus algoritmus (ld. 1. félév)
- X: x_1 x_2 … x_n, Y: y_1 y_2 … y_m, X-ből szeretnénk előállítani Y-t
- $d(x_i,y_j) = d(i,j) \geq 0$, $0 \leq i \leq n$, $0 \leq j \leq m$
- $d(0, 0) = 0$, $d(i, 0) = i$, $d(0, j) = j$
- $d(i, j) = \min(d(i, j-1) + 1, d(i-1, j) + 1, \ldots$

  $d(i-1, j-1) + (x_i=y_j) \; ? \; 0 : 1)$
- d(n, m) az optimális műveletsorozat összköltsége
- Ebből optimális műveletsorozat: visszakeresés

Kitöltött táblázat

-	0	E	X	E	C	U	T	I	O	N
0	0	1	2	3	4	5	6	7	8	9
I	1	1	2	3	4	5	6	6	7	8
N	2	2	2	3	4	5	6	7	7	7
T	3	3	3	3	4	5	5	6	7	8
E	4	3	4	3	4	5	6	6	7	8
N	5	4	4	4	4	5	6	7	7	7
T	6	5	5	5	5	5	5	6	7	8
I	7	6	6	6	6	6	6	5	6	7
O	8	7	7	7	7	7	7	6	5	6
N	9	8	8	8	8	8	8	7	6	5

Súlyozás

- Az eredeti algoritmusban a beszúrásnak, törlésnek, cserének nincs külön költsége
- De lehet: c_i, c_d, c_s (insertion, deletion, substitution)
  - Pl. 3/3/4, vagy 7/7/10
- $d(0, 0) = 0$, $d(i, 0) = ic_d$, $d(0, j) = jc_i$
- $d(i, j) = \min(d(i, j-1) + c_i, d(i-1, j) + c_d, \ldots$

  $d(i-1, j-1) + (x_i=y_j) \; ? \; 0 : c_s)$

Beszédfelismerési alkalmazás

- Mondatszintű (bemondásszintű) beszédfelismerést végzünk
  - …valamilyen módon (ld. Tóth László: Természetes nyelvi feldolgozás, beszédfelismerés c. MSc-tárgy)
- Ismert a mondat helyes (etalon) átirata
- A beszédfelismerő rendszerünknek is van egy szószintű kimenete (várhatóan nem pont ugyanannyi szóból áll)
- Kérdés: a kettő mennyire hasonlít? Mennyire pontosan találtuk el az etalon átiratot?

Beszédfelismerési alkalmazás

- **Etalon:** RÁADÁSUL AZ **ELEMZŐK** SZERINT EZ **A** KÜLÖNADÓ EGYÁLTALÁN NEM SEGÍTI MAJD AZ AMÚGY IS BAJBAN LÉVŐ KIS ÉS KÖZÉPVÁLLALKOZÁSOK PÉNZHEZ JUTÁSÁT
- **Felismert:** RÁADÁSUL AZ **ELEMZŐ** SZERINT EZ KÜLÖNADÓ EGYÁLTALÁN NEM SEGÍTI MAJD AZ AMÚGY IS BAJBAN LÉVŐ KIS ÉS KÖZÉPVÁLLALKOZÁSOK PÉNZHEZ JUTÁSÁT
- Accuracy = (N – S – D – I) / N
- N: etalonban lévő szavak száma, S: cserék száma, D: törlések száma, I: beszúrások száma
- Most: N = 21, S = 1, D = 1, I = 0, Acc= 19 / 21 = 90,5%