InfoLab

IC612: Data Warehousing and Data Mining

Lecture 3:
Similarity and Distances

Min-Soo Kim

DGIST 대구경북과학기술원
Daegu Gyeongbuk Institute of Science & Technology

Introduction

- **Many data mining applications require the determination of similar or dissimilar**
  - among objects, patterns, attributes, and events in the data
  - e.g., clustering, outlier detection, and classification

Given two objects O_1 and O_2, determine a value of the similarity $Sim(O_1, O_2)$ (or distance $Dist(O_1, O_2)$) between the two objects.

  - poor choices of the distance function can sometimes be disastrously misleading depending on the application domain
  - good distance function design is also crucial for type portability

- **Distance functions are highly sensitive to the data distribution, dimensionality, and data type**

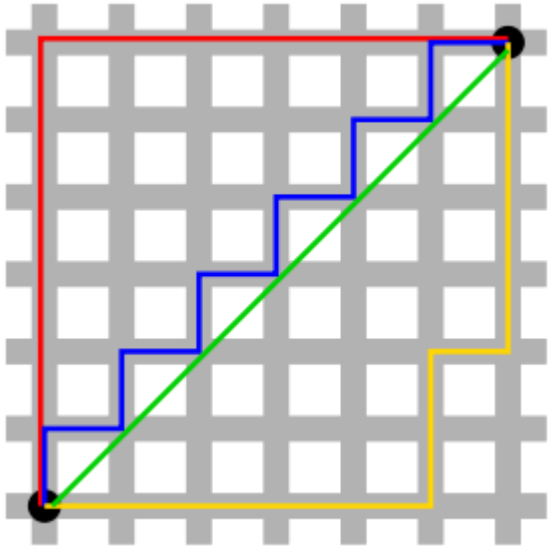
Quantitative Data

- **Most common distance function is the L_p-norm**
  - also called as Minkowski distance

$$\overline{X} = (x_1 \ldots x_d) \text{ and } \overline{Y} = (y_1 \ldots y_d)$$

$$Dist(\overline{X}, \overline{Y}) = \left(\sum_{i=1}^{d} |x_i - y_i|^p \right)^{1/p}$$

- **Manhattan distance (p=1)**
- **Euclidean distance (p=2)**
- **Supremum distance (p = ∞)**



Impact of Domain-specific Relevance

- **Which features are more important than others for a particular application?**
  - e.g., for a credit-scoring application, “salary” is much more relevant to the design of the distance function than “gender”

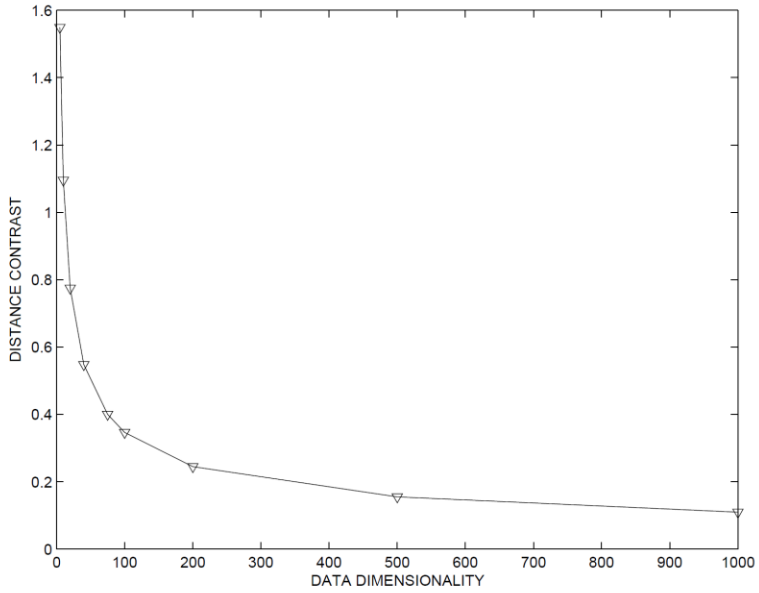
- **Generalized L_p-distance**
  - a coefficient a_i is associated with the i-th feature

$$Dist(\overline{X}, \overline{Y}) = \left(\sum_{i=1}^{d} a_i \cdot |x_i - y_i|^p \right)^{1/p}$$

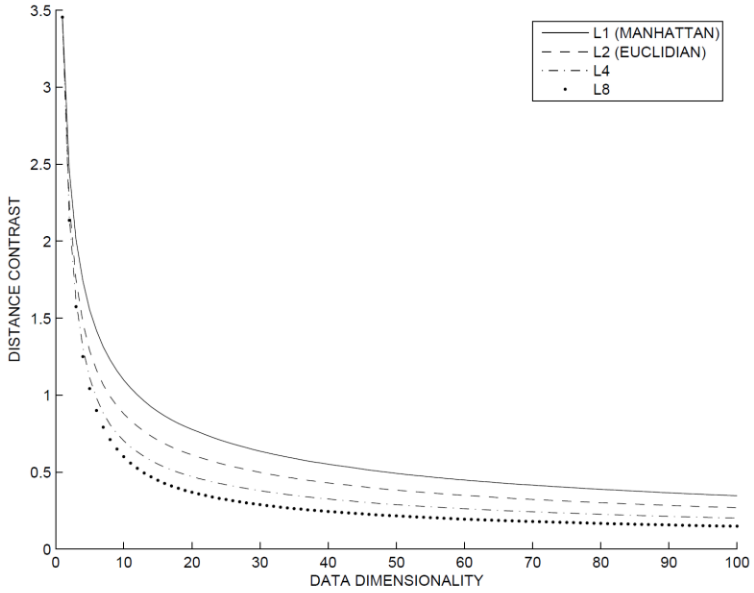
Impact of High Dimensionality

Curse of dimensionality

- many distance-based data mining applications lose their effectiveness as the dimensionality of the data increases
  - e.g., distance-based models of clustering, classification, and outlier detection are often *qualitatively* ineffective
- measure: distance contrast between the different data points
  - there is virtually no contrast with increasing dimensionality
  - direct use of the L_p-norm may not be effective with increasing d



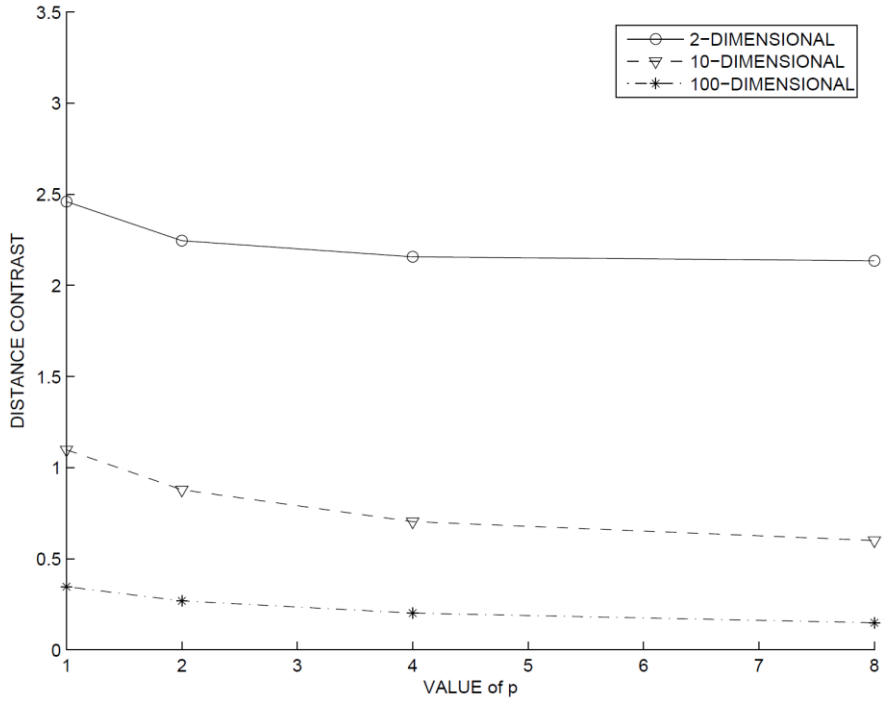
(a) Contrasts with dimensionality



(b) Contrasts with norms

Impact of Different L_p-norms

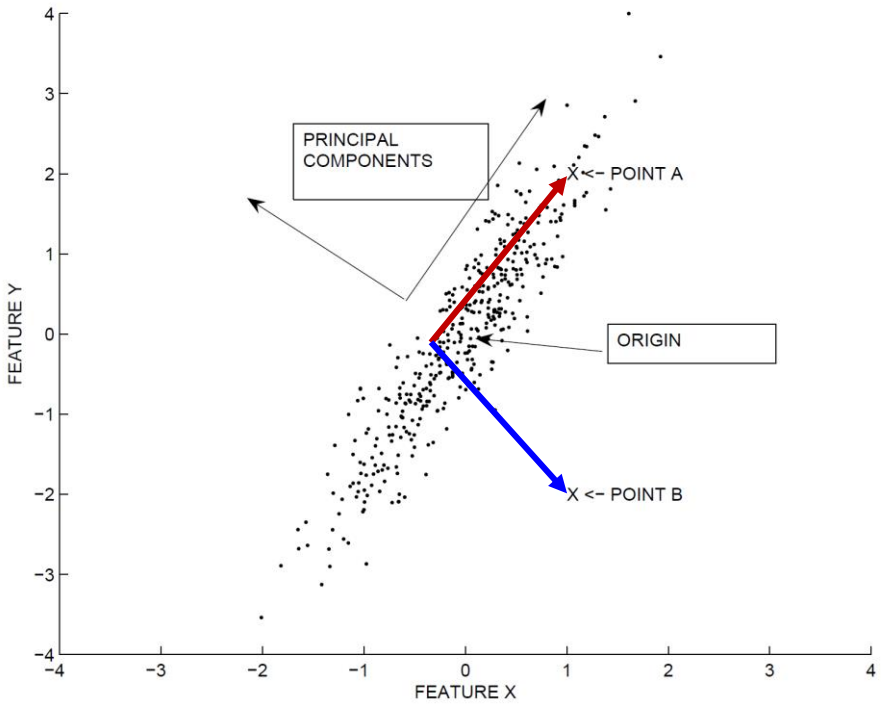
- **Many features are likely to be irrelevant in a typical high-dimensional data set**
  - e.g., for clustering diabetic patients based on medical records, blood glucose level feature are more important than other features
- **For larger p, irrelevant attributes are emphasized in L_∞**
  - e.g., for a 1000-dimensional application, if two objects have similar values on 999 attributes, they should be considered very similar
  - local similarity properties of the data are de-emphasized



Impact of Data Distribution

- **Should distances depend on the underlying data distribution of the remaining points in the data set?**

- **e.g., A and B are equidistant from the origin in any L_p-norm**
  - straight line from O to A follows a high variance direction
  - statistically, it is more likely for A to be away in red direction
  - but, it is much less likely for B to be so far away in blue direction
  - distance from O to A ought to be less than that of O to B



Mahalanobis distance

- **Mahalanobis distance**
  - two d-dimensional data points X and Y
  - Σ : d×d co-variance matrix of the data set

$$Maha(\overline{X},\overline{Y}) = \sqrt{(\overline{X}-\overline{Y})\Sigma^{-1}(\overline{X}-\overline{Y})^T}$$

  - normalizing the data on the basis of the inter-attribute correlations
  - Euclidean distance in a transformed (axes-rotated) data set after dividing each of the transformed coordinate values (PCA)
    - by the standard-deviation of the data along that direction
  - e.g., data point B will have a larger distance than data point A

Categorical Data

- **How can distances for categorical data be computed?**
  - no ordering exists among the discrete values

- **One possible solution**
  - categorical data can be transformed to numeric data with binarization
  - similarity functions (rather than distance functions) are adapted since binary vector is likely to be sparse
    - discrete values can be matched more naturally
    - $S(x_i, y_i) = 1$ when $x_i = y_i$, and 0 otherwise

$$Sim(\overline{X}, \overline{Y}) = \sum_{i=1}^{d} S(x_i, y_i)$$

Major drawback

- not account for the relative frequencies among different attributes
- e.g., “Normal” for 99% of the records, “Cancer” or “Diabetes” for 1% of the records
  - if two records have a “Normal” value for this variable, this does not provide statistically significant information about the similarity
  - if the two records have a matching “Cancer” or “Diabetes” value for this variable, it provides significant statistical evidence of similarity

Inverse Occurrence Frequency

- matches on unusual values of a categorical attribute should be weighted more heavily than values that appear frequently
- $p_k(x)$: probability that the k-th attribute takes on the value of x

$$S(x_i, y_i) = \begin{cases} 1/p_k(x_i)^2 & \text{if } x_i = y_i \\ 0 & \text{otherwise} \end{cases}$$

Text Similarity Measures

- **Multidimensional data under the "bag of words" model**
- **Applying L_p-norm**
  - do not adjust well to the varying length of the different documents
  - e.g., distance between two long documents always are larger than that between two short documents even if
    - the two long documents have many words in common
    - the short documents are completely disjoint
- **Cosine measure**
  - compute the angle between the two documents
  - insensitive to the absolute length of the document

$$\cos(\overline{X}, \overline{Y}) = \frac{\sum_{i=1}^{d} x_i \cdot y_i}{\sqrt{\sum_{i=1}^{d} x_i^2} \cdot \sqrt{\sum_{i=1}^{d} y_i^2}}$$

Inverse document frequency (IDF)

- if two documents match on an uncommon word, it is more indicative of similarity
- n_i: number of documents in which the i-th word occurs
- n: number of documents in the collection

$$id_i = \log(n/n_i)$$

Improved cosine measure

- $f(x_i)$: damping function

$$f(x_i) = \sqrt{x_i}$$
$$f(x_i) = \log(x_i)$$

- $h(x_i)$: normalized frequency for the i-th word

$$h(x_i) = f(x_i) \cdot id_i$$

$$\cos(\overline{X}, \overline{Y}) = \frac{\sum_{i=1}^{d} h(x_i) \cdot h(y_i)}{\sqrt{\sum_{i=1}^{d} h(x_i)^2} \cdot \sqrt{\sum_{i=1}^{d} h(y_i)^2}}$$

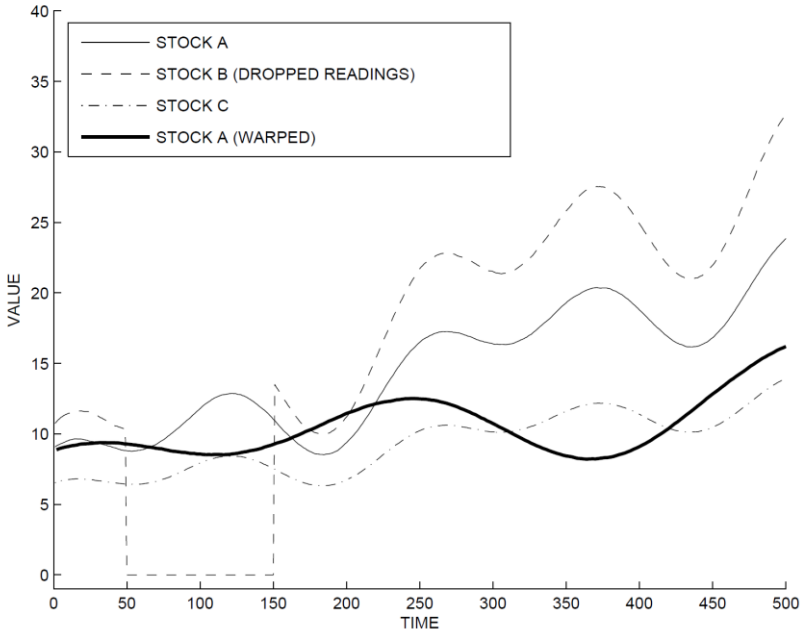
Binary and Set Data

- **Jaccard coefficient**

$$J(\overline{X}, \overline{Y}) = \frac{\sum_{i=1}^{d} x_i \cdot y_i}{\sum_{i=1}^{d} x_i^2 + \sum_{i=1}^{d} y_i^2 - \sum_{i=1}^{d} x_i \cdot y_i} = \frac{|S_X \cap S_Y|}{|S_X \cup S_Y|}$$

Temporal Similarity Measures

- **Temporal data**
  - a single contextual attribute, one or more behavioral attributes
  - continuous time series (c.f., discrete sequences)
- **Several distortion factors**
  - Behavioral attribute scaling and translation: absolute values may be very different both in terms of the mean and the standard deviation
  - Temporal attribute translation
  - Temporal attribute scaling: stretch or compress along the temporal axis to allow more effective matching (time warping)
  - Non-contiguity in matching



L_p-norms

- **Appling to wavelet transformations of the time series**
  - Euclidean metrics are invariant to axis rotation

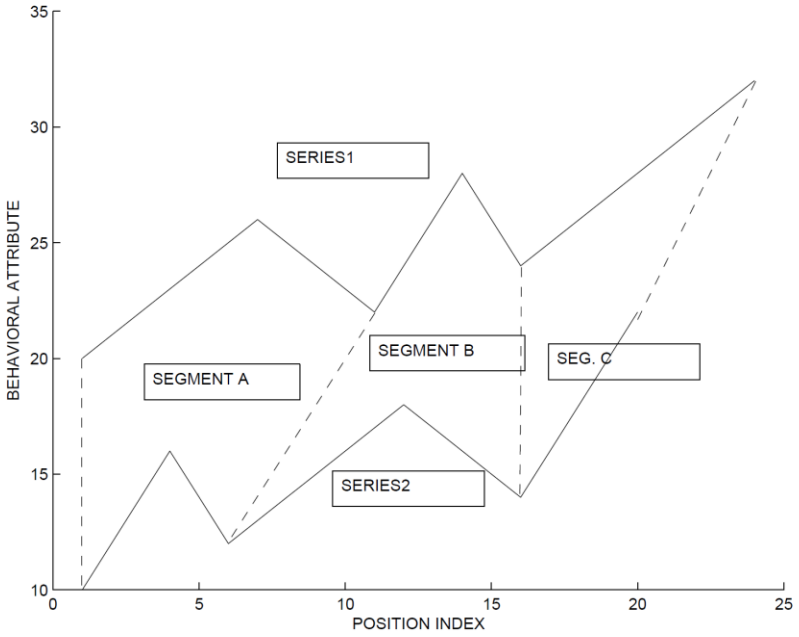
$$\overline{X} = (x_1 \ldots x_n) \text{ and } \overline{Y} = (y_1 \ldots y_n)$$

$$Dist(\overline{X}, \overline{Y}) = \left(\sum_{i=1}^{n} |x_i - y_i|^p \right)^{1/p}$$

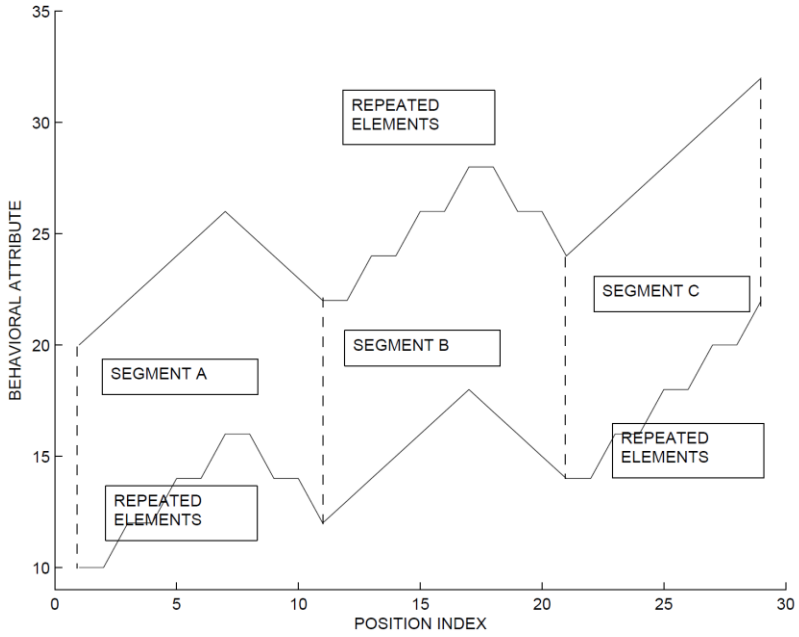
- **Major problems**
  - designed for time series of equal length
  - cannot address distortions on the temporal (contextual) attributes

Dynamic Time Warping (DTW)

- **Stretching the series along the time-axis**
  - in a varying (or dynamic) way over different portions to enable more effective matching
  - adapted from the field of speech recognition, where time warping was deemed necessary to match different speaking speeds



(a) Original series

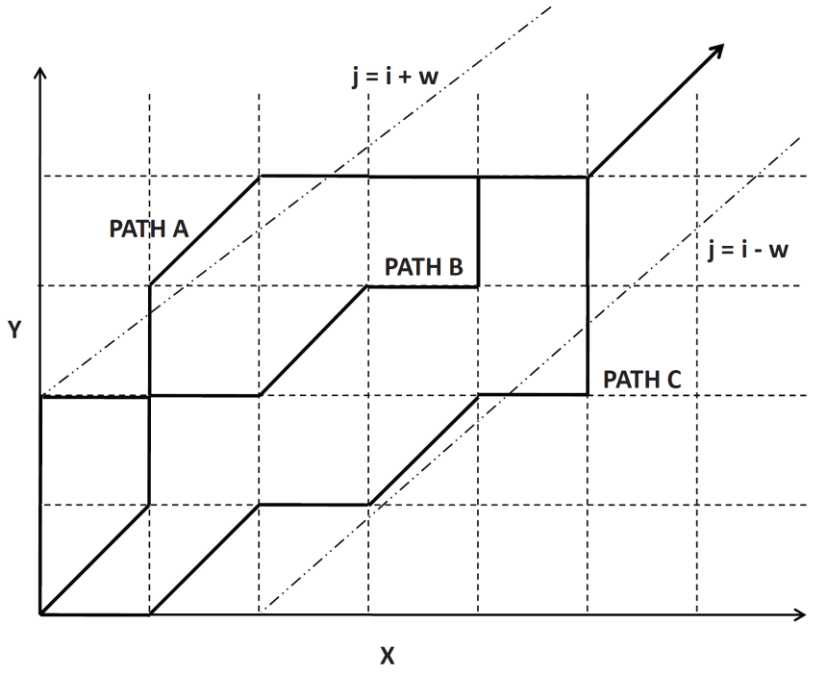


(b) Warped series

- **Artificially creating two series of the same length that have a one-to-one mapping between them**
  - goal: mapping in an *optimal* way to minimize the dynamic time warping distance (by using dynamic programming)
  - DTW(i, j) : optimal distance between the first i and first j elements

$$DTW(i,j) = distance(x_i, y_j) + \min \begin{cases} DTW(i, j-1) & \text{repeat } x_i \\ DTW(i-1, j) & \text{repeat } y_j \\ DTW(i-1, j-1) & \text{repeat neither} \end{cases}$$

  - w: window constraint
    - imposes a minimum level of positional alignment between matched elements
    - e.g., paths B and C do not need to be computated



Discrete Sequence Similarity Measures

- **Edit distance (Levenshtein distance)**
  - optimal “effort” (or cost) required to transform one sequence into another by using a series of edit operations
  - edit operations : symbol insertions, deletions, and replacements with specific costs
  - in many models (e.g., biology)
    - replacements have higher cost than insertions or deletions
    - insertions and deletions have the same cost
  - computation of the optimal cost requires dynamic programming

$$Edit(i,j) = \min \begin{cases} Edit(i-1,j) + \text{Deletion Cost} \\ Edit(i,j-1) + \text{Insertion Cost} \\ Edit(i-1,j-1) + I_{ij} \cdot (\text{Replacement Cost}) \end{cases}$$

Longest Common Subsequence (LCS)

- **Subsequence of a sequence**
  - a set of symbols drawn from the sequence in the same order as the original sequence (maybe not contiguous)
  - e.g., "abcde" is a subsequence of "agbfcgdhei"
  - c.f., substring (contiguous)
  - LCS is a similarity function (higher values indicate greater similarity)

$$\overline{X} = (x_1 \ldots x_m) \text{ and } \overline{Y} = (y_1 \ldots y_n)$$

- **LCS(i, j) : optimal LCS between the first i symbols and j symbols**

$$LCSS(i,j) = \max \begin{cases} LCSS(i-1, j-1) + 1 & \text{only if } x_i = y_j \\ LCSS(i-1, j) & \text{otherwise (no match on } x_i) \\ LCSS(i, j-1) & \text{otherwise (no match on } y_j) \end{cases}$$

Graph Similarity Measures

- **Similarity between two nodes in a single graph**
  - Structural distance-based measure (based on shortest path distance)
  - Random walk-based similarity (SimRank)

- **Similarity between two graphs**
  - e.g., comparing chemical compounds
  - determining whether two graphs are identical is extremely challenging
  - formally, graph isomorphism problem (NP-hard)
  - several measures
    - Maximum common subgraph distance: two graphs contain a large subgraph in common
    - Substructure-based similarity
    - Graph-edit distance
    - Graph kernels

InfoLab

Dynamic Programming III

Min-Soo Kim

DGIST

DGIST 대구경북과학기술원 Daegu Gyeongbuk Institute of Science & Technology

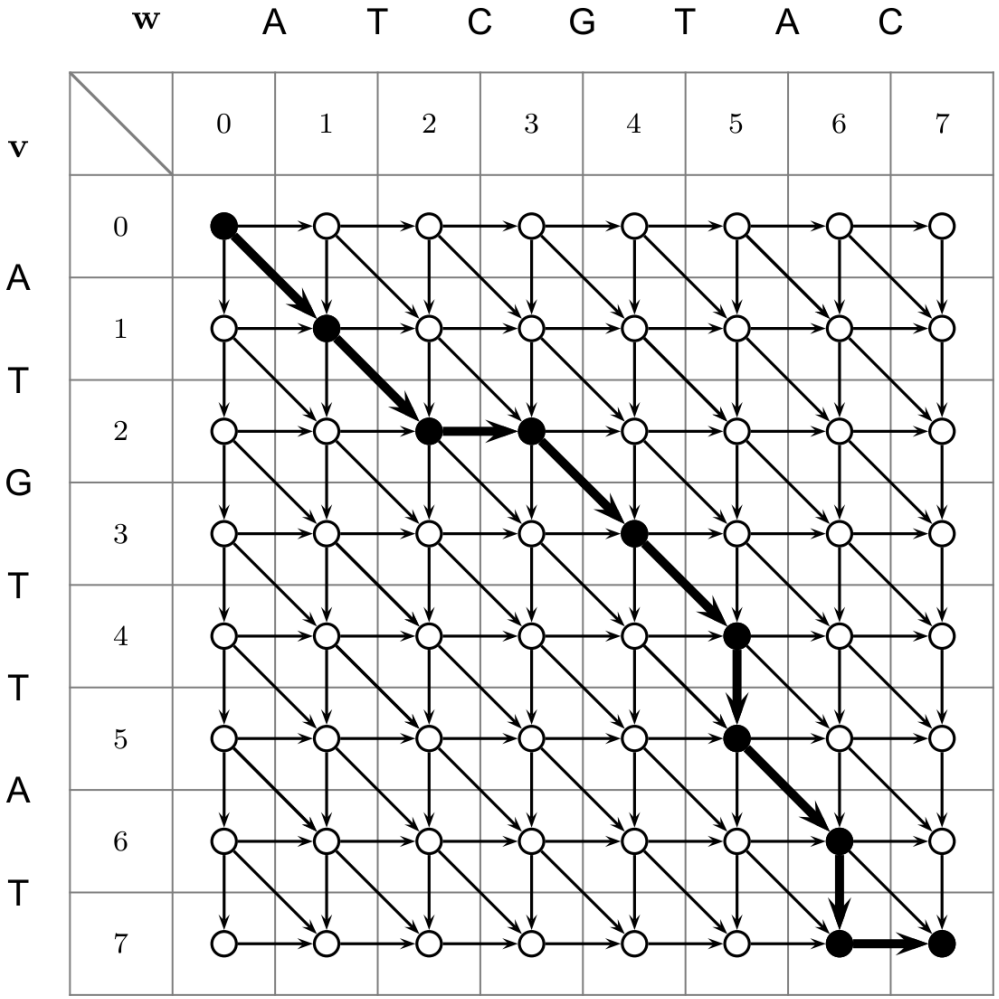
Alignment

- **Alignment of v (of n characters) and w (of m characters)**
  - two-row matrix s.t. the first row contains the characters of v in order while the second row contains the characters of w in order
  - spaces may be interspersed throughout the strings in different places
  - alignment may have at most n+m columns

A	T	-	G	T	T	A	T	-
A	T	C	G	T	-	A	-	C

$$\begin{array}{cccccccccc} & \mathsf{A} & \mathsf{T} & - & \mathsf{G} & \mathsf{T} & \mathsf{T} & \mathsf{A} & \mathsf{T} & - \\ \binom{0}{0} & \binom{1}{1} & \binom{2}{2} & \binom{2}{3} & \binom{3}{4} & \binom{4}{5} & \binom{5}{5} & \binom{6}{6} & \binom{7}{6} & \binom{7}{7} \\ & \mathsf{A} & \mathsf{T} & \mathsf{C} & \mathsf{G} & \mathsf{T} & - & \mathsf{A} & - & \mathsf{C} \end{array}$$

		0	1	2	2	3	4	5	6	7	7
v	=		A	T	-	G	T	T	A	T	-
			|	|		|	|		|		
w	=		A	T	C	G	T	-	A	-	C
		0	1	2	3	4	5	5	6	6	7



General scoring

- **Restrictive scoring (in the LCS problem)**
  - score = 1 for match
  - no penalty for **indel** (insert/delete)

- **General scoring**
  - k-letter alphabet $\mathbb{A}$ (including the gap character "–")
  - (k+1) x (k+1) arbitrary **scoring matrix** δ
  - score of the column (x, y) in the alignment : $\delta(x, y)$
  - scoring of mismatches and indels in the alignment

- **k : typically**
  - 4 for nucleotides (DNA), or
  - 20 for amino acids (protein)

Global sequence alignment

- **Problem: Find the best alignment between two strings under a given scoring matrix**
- **Input: strings v, w and a scoring matrix δ**
- **Output: An alignment of v and w whose score is maximal among all possible alignments of v and w**

Score of optimal alignment

- Recurrence for the score $s_{i,j}$ of an optimal alignment between the i-th prefix of v and j-th prefix of w

$$s_{i,j} = \max \begin{cases} s_{i-1,j} + \delta(v_i, -) \\ s_{i,j-1} + \delta(-, w_j) \\ s_{i-1,j-1} + \delta(v_i, w_j) \end{cases}$$

- **Mismatch**: penalized by constant $-\mu$
- **Indel**: penalized by constant $-\sigma$
- **Match**: rewarded with +1

Recurrence for score $s_{i,j}$

- **Resulting score**

$$\#matches - \mu \cdot \#mismatches - \sigma \cdot \#indels$$

- **Recurrence**

$$s_{i,j} = \max \begin{cases} s_{i-1,j} - \sigma \\ s_{i,j-1} - \sigma \\ s_{i-1,j-1} - \mu, \text{ if } v_i \neq w_j \\ s_{i-1,j-1} + 1, \text{ if } v_i = w_j \end{cases}$$

- **LCS problem = Global alignment problem with $\mu = \sigma = 0$**

Scoring alignment

- **For nucleotides**

	A	C	G	T	–
A	+1	−1	−1	−1	−1
C	−1	+1	−1	−1	−1
G	−1	−1	+1	−1	−1
T	−1	−1	−1	+1	−1
-	−1	−1	−1	−1	N/D

- **For amino acids**
  - very few matches but still represent biologically adequate alignments (in contrast to nucleotides)

Scoring alignments

- **Amino acids substitutions**
  - some amino acid substitutions are commonly found throughout the process of molecular evolution
  - but, others are rare
  - e.g., Asn, Asp, Glu, and Ser : most “mutable” amino acids
  - e.g., Cys and Trp : least mutable

알라닌	Alanine	A	Ala
시스테인	Cysteine	C	Cys
아스파르트산	Aspartic acid	D	Asp
글루탐산	Glutamic acid	E	Glu
페닐알라닌	Phenylalanine	F	Phe
글라이신	Glycine	G	Gly
히스티딘	Histidine	H	His
아이소류신	Isoleucine	I	Ile
라이신	Lysine	K	Lys
류신	Leucine	L	Leu
메티오닌	Methionine	M	Met
아스파라긴	Asparagine	N	Asn
피롤라이신	Pyrrolysine	O	Ply
프롤린	Proline	P	Pro
글루타민	Glutamine	Q	Gln
아르기닌	Arginine	R	Arg
세린	Serine	S	Ser
트레오닌	Threonine	T	Thr
셀레노시스테인	Selenocysteine	U	Sec
발린	Valine	V	Val
트립토판	Tryptophan	W	Trp
타이로신	Tyrosine	Y	Tyr

Scoring matrix (substitution matrix)

- **Frequency with which amino acid x replaces amino acid y in evolutionarily related sequences**
- **δ(i,j): how often AA i substitutes AA j in the alignments of related protein sequences**
- **Best scoring matrices to compare two proteins depends on how similar these organisms are**
- **Point Accepted Mutations (PAM)**
- **BLOck SUbstitution (BLOSUM)**

PAM1

	A	R	N	D	C	Q	E	G	H	I	L	K	M	F	P	S	T	W	Y	V
A	9867	2	9	10	3	8	17	21	2	6	4	2	6	2	22	35	32	0	2	18
R	1	9913	1	0	1	10	0	0	10	3	1	19	4	1	4	6	1	8	0	1
N	4	1	9822	36	0	4	6	6	21	3	1	13	0	1	2	20	9	1	4	1
D	6	0	42	9859	0	6	53	6	4	1	0	3	0	0	1	5	3	0	0	1
C	1	1	0	0	9973	0	0	0	1	1	0	0	0	0	1	5	1	0	3	2
Q	3	9	4	5	0	9876	27	1	23	1	3	6	4	0	6	2	2	0	0	1
E	10	0	7	56	0	35	9865	4	2	3	1	4	1	0	3	4	2	0	1	2
G	21	1	12	11	1	3	7	9935	1	0	1	2	1	1	3	21	3	0	0	5
H	1	8	18	3	1	20	1	0	9912	0	1	1	0	2	3	1	1	1	4	1
I	2	2	3	1	2	1	2	0	0	9872	9	2	12	7	0	1	7	0	1	33
L	3	1	3	0	0	6	1	1	4	22	9947	2	45	13	3	1	3	4	2	15
K	2	37	25	6	0	12	7	2	2	4	1	9926	20	0	3	8	11	0	1	1
M	1	1	0	0	0	2	0	0	0	5	8	4	9874	1	0	1	2	0	0	4
F	1	1	1	0	0	0	0	1	2	8	6	0	4	9946	0	2	1	3	28	0
P	13	5	2	1	1	8	3	2	5	1	2	2	1	1	9926	12	4	0	0	2
S	28	11	34	7	11	4	6	16	2	2	1	7	4	3	17	9840	38	5	2	2
T	22	2	13	4	1	3	2	2	1	11	2	8	6	1	5	32	9871	0	2	9
W	0	2	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	9976	1	0
Y	1	0	3	0	3	0	1	0	4	1	1	0	0	21	0	1	1	2	9945	1
V	13	2	1	1	3	2	2	3	3	57	11	1	17	1	3	2	10	0	2	9901

PAM250

	A	R	N	D	C	Q	E	G	H	I	L	K	M	F	P	S	T	W	Y	V
A	13	6	9	9	5	8	9	12	6	8	6	7	7	4	11	11	11	2	4	9
R	3	17	4	3	2	5	3	2	6	3	2	9	4	1	4	4	3	7	2	2
N	4	4	6	7	2	5	6	4	6	3	2	5	3	2	4	5	4	2	3	3
D	5	4	8	11	1	7	10	5	6	3	2	5	3	1	4	5	5	1	2	3
C	2	1	1	1	52	1	1	2	2	2	1	1	1	1	2	3	2	1	4	2
Q	3	5	5	6	1	10	7	3	7	2	3	5	3	1	4	3	3	1	2	3
E	5	4	7	11	1	9	12	5	6	3	2	5	3	1	4	5	5	1	2	3
G	12	5	10	10	4	7	9	27	5	5	4	6	5	3	8	11	9	2	3	7
H	2	5	5	4	2	7	4	2	15	2	2	3	2	2	3	3	2	2	3	2
I	3	2	2	2	2	2	2	2	2	10	6	2	6	5	2	3	4	1	3	9
L	6	4	4	3	2	6	4	3	5	15	34	4	20	13	5	4	6	6	7	13
K	6	18	10	8	2	10	8	5	8	5	4	24	9	2	6	8	8	4	3	5
M	1	1	1	1	0	1	1	1	1	2	3	2	6	2	1	1	1	1	1	2
F	2	1	2	1	1	1	1	1	3	5	6	1	4	32	1	2	2	4	20	3
P	7	5	5	4	3	5	4	5	5	3	3	4	3	2	20	6	5	1	2	4
S	9	6	8	7	7	6	7	9	6	5	4	7	5	3	9	10	9	4	4	6
T	8	5	6	6	4	5	5	6	4	6	4	6	5	3	6	8	11	2	3	6
W	0	2	0	0	0	0	0	0	1	0	1	0	0	1	0	1	0	55	1	0
Y	1	1	2	1	3	1	1	1	3	2	2	1	2	15	1	2	2	3	31	2
V	7	4	4	4	4	4	4	4	5	4	15	10	4	10	5	5	5	72	4	17

PAM250 (log-odds)

	A	R	N	D	C	Q	E	G	H	I	L	K	M	F	P	S	T	W	Y	V
A	2	-2	0	0	-2	0	0	1	-1	0	-2	-1	-1	-3	1	1	1	-6	-4	0
R	-2	6	0	-1	-4	1	-1	-3	2	-2	-3	3	0	-4	0	0	-1	2	-5	-2
N	0	0	2	2	-3	1	2	1	2	-2	-3	1	-2	-3	0	1	1	-4	-2	-2
D	0	-1	2	4	-5	2	3	1	1	-2	-4	0	-3	-5	-1	0	0	-7	-4	-2
C	-2	-3	-4	-5	12	-5	-5	-4	-3	-3	-6	-5	-5	-4	-2	0	-2	-8	0	-2
Q	0	1	1	2	-5	4	2	-1	3	-2	-2	1	-1	-4	0	-1	-1	-5	-4	-2
E	0	-1	1	3	-5	2	4	0	1	-2	-3	0	-2	-5	0	0	0	-7	-4	-2
G	1	-3	0	1	-3	-1	0	5	-2	-2	-4	-2	-3	-5	0	1	0	-7	-5	-1
H	-1	1	1	1	-3	3	0	-3	6	-3	-3	0	-3	-2	0	-1	-1	-3	0	-3
I	-1	-2	-2	-2	-2	-2	-2	-3	-3	4	2	-2	2	1	-2	-1	0	-5	-1	4
L	-2	-3	-3	-4	-6	-2	-3	-4	-2	2	6	-2	4	2	-2	-3	-2	-2	-1	2
K	-1	3	1	0	-5	1	0	-2	0	-2	-3	5	0	-5	-1	0	0	-4	-5	-2
M	-1	-1	-2	-3	-5	-1	-2	-3	-2	2	4	1	6	0	-2	-2	0	-4	-3	2
F	-3	-4	-3	-5	-4	-4	-5	-4	-2	1	2	-5	0	9	-5	-3	-3	0	7	-1
P	1	0	0	-1	-3	0	0	0	0	-2	-2	-1	-2	-4	6	1	0	-6	-5	-1
S	2	1	2	1	1	0	1	2	0	-1	-2	1	-1	-2	2	2	2	-2	-2	0
T	0	-2	0	-1	-3	-2	-1	-1	-2	-1	-2	-1	-1	-4	0	1	2	-6	-4	0
W	-6	2	-5	-7	-7	-6	-7	-7	-5	-6	-7	-4	-6	1	-6	-2	-5	17	1	-8
Y	-3	-5	-2	-4	1	-4	-4	-5	0	-1	-1	-5	-2	7	-5	-3	-3	0	10	-2
V	0	-2	-2	-2	-2	-2	-2	-2	-2	4	2	-2	2	-1	-1	-1	0	-6	-3	4

Issue of global sequence alignment

- **Global alignment problem**
  - seek similarities between two entire strings
  - useful when the similarity extends over their entire length
  - e.g., protein sequences from the same protein family

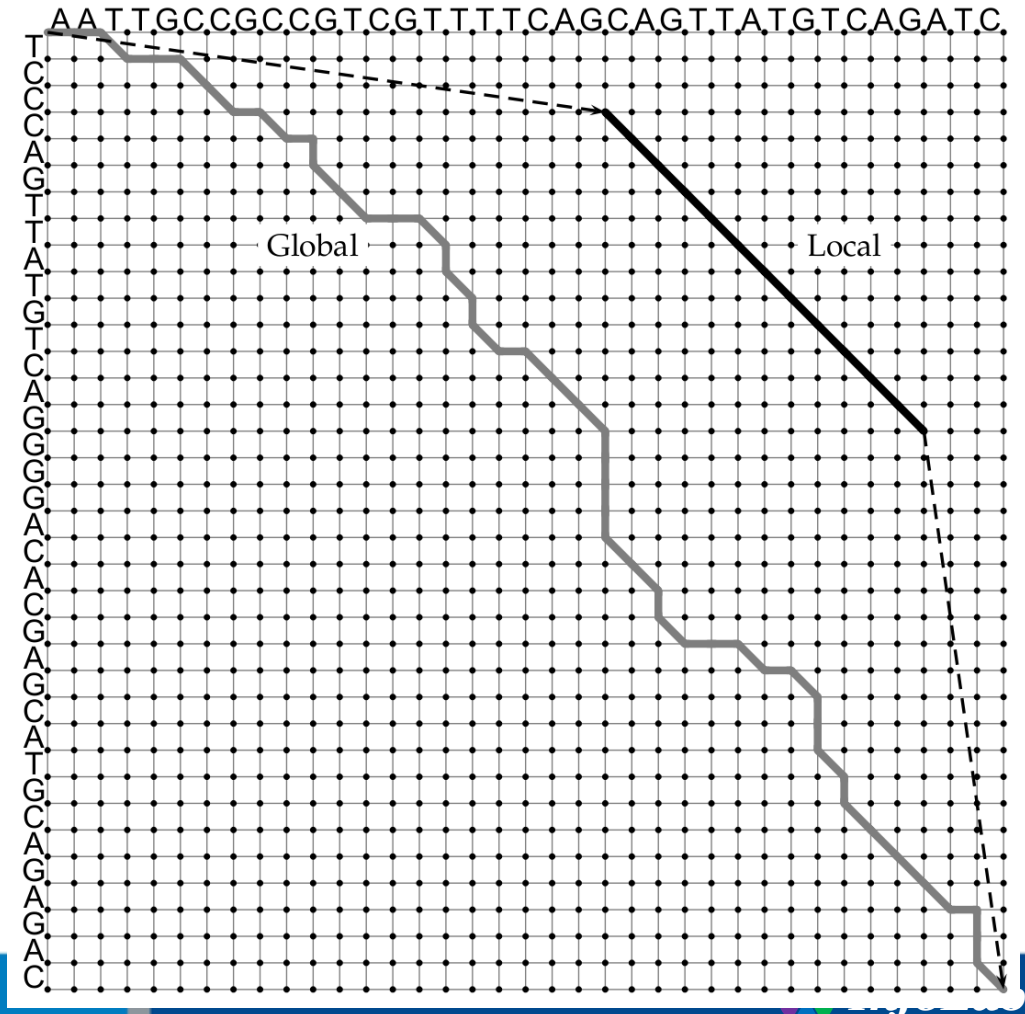
- **In many biological applications**
  - score of an alignment between two substrings of v and w
    might actually be larger than
    score of an alignment between the entireties of v and w
  - global alignment cannot find conserved area due to so many indels

global

```
--T--CC-C-AGT--TATGT-CAGGGGACACG--A-GCATGCAGA-GAC
  |  || |  ||  | | | |||    || |  | |  | ||||   |
AATTGCCGCC-GTCGT-T-TTCAG----CA-GTTATG--T-CAGAT--C
```

local

```
                        tccCAGTTATGTCAGgggacacgagcatgcagagac
                           ||||||||||||
aattgccgccgtcgttttcagCAGTTATGTCAGatc
```

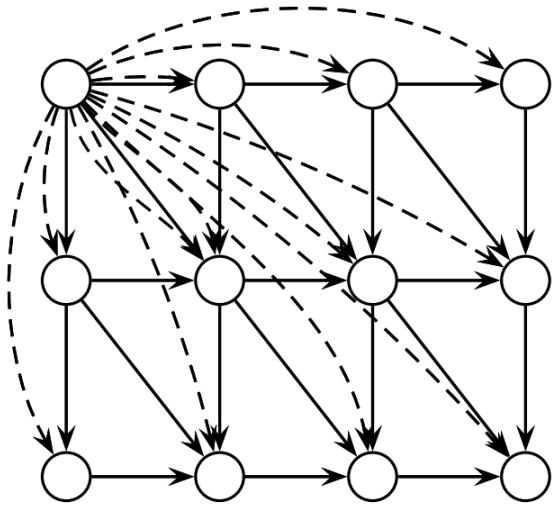


Local sequence alignment

- **Motivation: biologically significant similarities are present in certain parts of DNA fragments**
- **Problem: Find the best local alignment between two strings**
  - maximize the alignment score over all substrings of v and w
- **Input: strings v and w and a scoring matrix δ**
- **Output: substrings of v and w whose global alignment is maximal**
  - among all global alignments of all substrings of v and w

Smith-Waterman algorithm

- **Introducing edges of weight 0 (dashed lines) from (0,0) to every other vertex**
  - initialize DP table with zeros
  - no negative cells



- **Recurrence**

$$s_{i,j} = \max \begin{cases} 0 \\ s_{i-1,j} + \delta(v_i, -) \\ s_{i,j-1} + \delta(-, w_j) \\ s_{i-1,j-1} + \delta(v_i, w_j) \end{cases}$$

Find optimal local alignment

- Locate the highest value in DP table
- Trace back until a zero entry

Q	U	E	V	I	V	A	L	A	S	V	E	G	A	S
R	R	R	|	|	|	|	R	R	R	R	R	R	R	R
-	-	-	V	I	V	A	D	A	-	V	-	-	I	S.

		Q	U	E	V	I	V	A	L	A	S	V	E	G	A	S
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
V	0	0	0	0	**1**	0	1	0	0	0	0	1	0	0	0	0
I	0	0	0	0	0	**2**	1	0	0	0	0	0	0	0	0	0
V	0	0	0	0	1	1	**3**	2	1	0	0	1	0	0	0	0
A	0	0	0	0	0	0	2	**4**	3	2	1	0	0	0	1	0
D	0	0	0	0	0	0	1	3	3	2	1	0	0	0	0	0
A	0	0	0	0	0	0	0	2	2	4	3	2	1	0	1	0
V	0	0	0	0	1	0	1	1	1	3	3	4	3	2	1	0
I	0	0	0	0	0	2	1	0	0	2	2	3	3	2	1	0
S	0	0	0	0	0	1	1	0	0	1	3	2	2	2	1	2

Gap penalties

- **Mutations : usually caused by errors in DNA replication**
  - nature frequently deletes or inserts entire substrings as a unit
  - common evolutionary events

- **Gap (in alignment) : contiguous sequence of spaces in one of the rows**
  - penalizing a gap of length x as $-\sigma x$ is unusual punishment
  - many practical algorithms use affine gap penalty functions

- **Affine gap penalty: $-\sigma - (x-1)\rho$, where $\sigma > \rho$**
  - σ : gap-open penalty
  - ρ : gap-extension penalty

Example: BLOSUM50

	H	E	A	G	A	W	G	H	E	E
P	−2	−1	−1	−2	−1	−4	−2	−2	−1	−1
A	−2	−1	**5**	0	**5**	−3	0	−2	−1	−1
W	−3	−3	−3	−3	−3	**15**	−3	−3	−3	−3
H	**10**	0	−2	−2	−2	−3	−2	**10**	0	0
E	0	**6**	−1	−3	−1	−3	−3	0	**6**	**6**
A	−2	−1	**5**	0	**5**	−3	0	−2	−1	−1
E	0	**6**	−1	−3	−1	−3	−3	0	**6**	**6**

Example : global alignment

		H	E	A	G	A	W	G	H	E	E
	0 ←	**–8** ←	**–16** ↖	–24 ←	–32 ←	–40 ←	–48 ←	–56 ←	–64 ←	–72 ←	–80
P	–8	–2	–9	**–17** ←	**–25** ↖	–33 ←	–42 ←	–49 ←	–57	–65	–73
A	–16	–10	–3	–4 ←	–12	**–20** ↖	–28 ←	–36 ←	–44 ←	–52 ←	–60
W	–24	–18	–11	–6	–7	–15	**–5** ←	**–13** ↖	–21 ←	–29 ←	–37
H	–32	–14	–18	–13	–8	–9	–13	–7	**–3** ↖	–11 ←	–19
E	–40	–22	–8 ←	–16	–16	–9	–12	–15	–7	**3**	–5
A	–48	–30	–16	–3 ←	–11	–11	–12	–12	–15	**–5** ↖	2
E	–56	–38	–24	–11	–6	–12	–14	–15	–12	–9	**1**

```
HEAGAWGHE-E
--P-AW-HEAE
```

gap penalty = -8

Example : local alignment

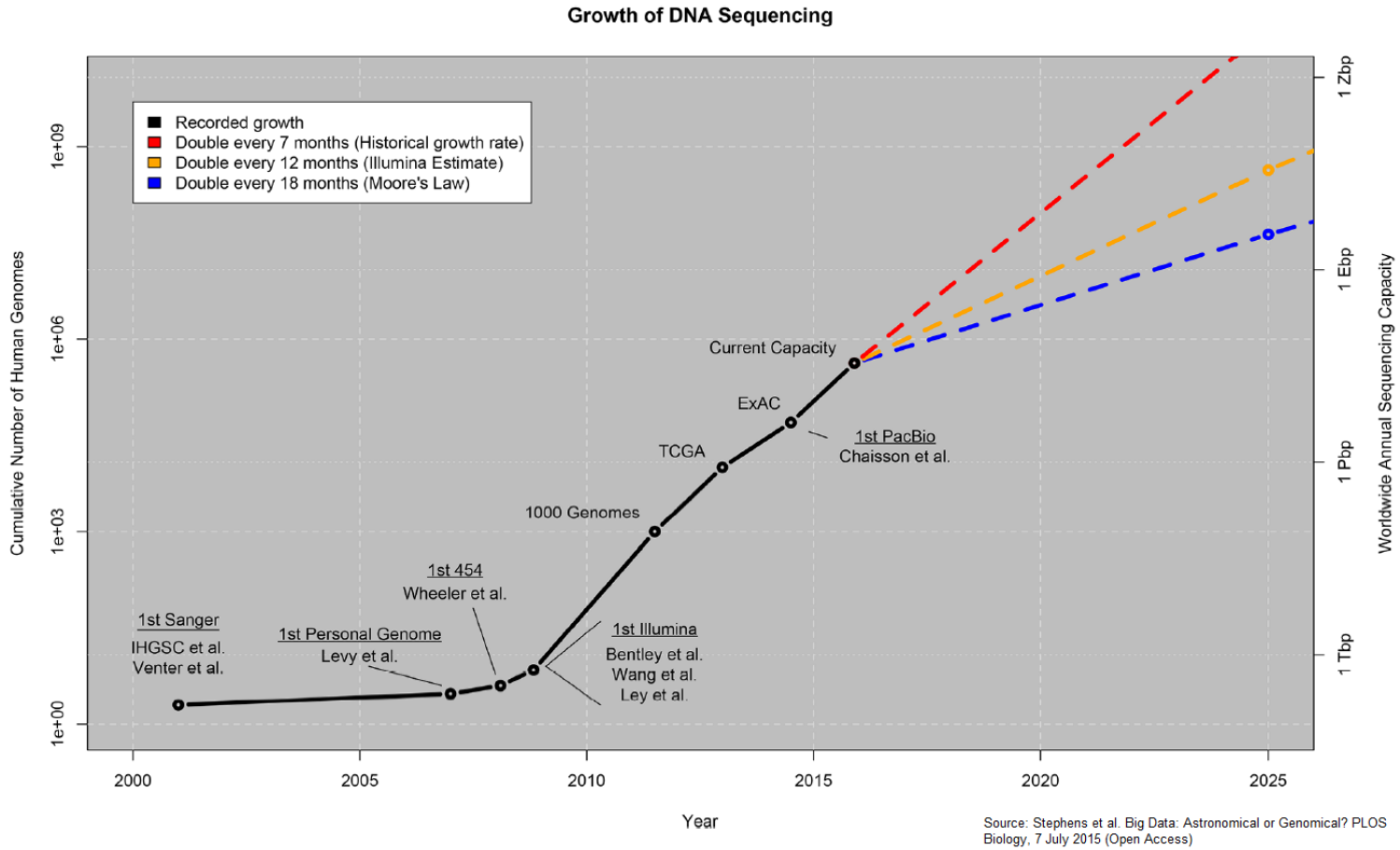
		H	E	A	G	A	W	G	H	E	E
	0	0	0	0	0	0	0	0	0	0	0
P	0	0	0	0	0	0	0	0	0	0	0
A	0	0	0	5	0	**5**	0	0	0	0	0
W	0	0	0	0	2	0	**20**	**12**	4	0	0
H	0	10	2	0	0	0	12	18	**22**	14	6
E	0	2	16	8	0	0	4	10	18	**28**	20
A	0	0	8	21	13	5	0	4	10	20	27
E	0	0	6	13	18	12	4	0	4	16	26

AWGHE

AW-HE

BLAST: fast approximate local alignment

- **Motivation: high computational cost of Smith-Waterman**
  - O(mn) is still high compared with the amount of DNA sequence data



Basic Local Alignment Search Tool (BLAST)

- **Quickly find the regions of high similarity**
- **Approximate alignment algorithm**
  - not guarantee finding the maximum scoring alignments
- **Find islands of similarity without gaps (segment pairs)**

Maximal segment pair (MSP)

- **Given two string s and t**

- **Segment pair**
  - pair of substrings of two sequences s and t
  - equal length
  - aligned without gaps
  - alignment score (without spaces) cannot be improved by extending it or shortening it

- **Maximal segment pair**
  - segment pair with the maximum score over all segment pairs

BLAST algorithm

- **Find all the sequences that when paired with the query contain an MSP whose score is above a threshold θ**
  - θ : chosen based on statistical considerations

- **Steps**
  - precompute an index of all length *l* words and their position in DB
  - identify a short segment pair (hotspots, or hits)
  - extend it (as seed) in both directions until the score drops below θ

InfoLab

Thank you!

DGIST 대구경북과학기술원
Daegu Gyeongbuk Institute of Science & Technology