Development of Complex Curricula for Molecular Bionics and Infobionics Programs within a consortial* framework**

Consortium leader

PETER PAZMANY CATHOLIC UNIVERSITY

Consortium members

SEMMELWEIS UNIVERSITY, DIALOG CAMPUS PUBLISHER

The Project has been realised with the support of the European Union and has been co-financed by the European Social Fund ***

**Molekuláris bionika és Infobionika Szakok tananyagának komplex fejlesztése konzorciumi keretben

***A projekt az Európai Unió támogatásával, az Európai Szociális Alap társfinanszírozásával valósul meg.

INTRODUCTION TO BIOINFORMATICS

(BEVEZETÉS A BIOINFORMATIKÁBA)

CHAPTER 3

Sequence Alignment Algorithms

(Szekvencia illesztési algoritmusok)

András Budinszky

Sequence Alignment

It is a process of comparing two (pairwise sequence alignment) or more (multiple sequence alignment) DNA or protein sequences.

The sequences are arranged to discover similarities that could show a functional, structural or evolutionary relationships between the sequences.

Similarity means a degree of match at corresponding positions of the sequences.

Similarity is usually a consequence of homology but it could occur by chance (when comparing short sequences).

Types of Pairwise Alignment

Global Alignment:

It attempts to align every position in the entire sequences and determine the measure of their similarity from end to end in each sequence.

It is usually used with sequences of approximately the same length.

Local Alignment:

It attempts to align sections of the sequences ("islands", conserved regions) with significant similarity.

It can be used for sequences of quite different length.

Methods for Pairwise Alignment

- Dot plot (matrix) analysis
- Dynamic programming algorithm
- Word or k-tuple methods

Note: Each method has its strengths and weaknesses, and all three pairwise methods have difficulty with highly repetitive sequences, especially if the number of repetitions differ.

Dot Plot (matrix) Analysis

It is a graphical method.

It is the simplest one and should be the primary method considered for pairwise sequence alignment.

It creates a two-dimensional matrix.

One of the sequences is written along the top row and the other along the leftmost column of the matrix.

A dot is placed at any point where the characters match and the rest of the points are left blank.

Matching sections of the sequences are shown as diagonals of dots.

It works best if it uses value thresholds.

Dot Matrix Programs

A number of them available:

DOTTER (http://sonnhammer.sbc.su.se/Dotter.html) with interactive features

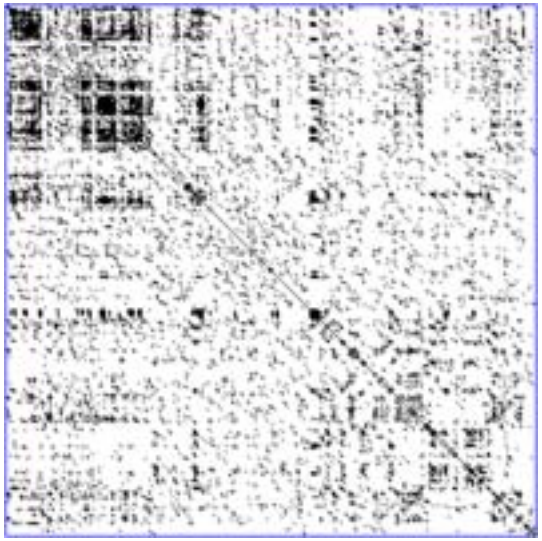
COMPARE and DOTPOLT (Genetics Computer Group)

EMBOSS suite (http://emboss.sourceforge.net/):

- dotmatcher (align sequences using a scoring matrix)
- dottup (finds common words in sequences)
- dotplot (finds common patterns in sequences)

Finding sequence repeats

A special use of dot matrix: aligning a sequence with itself:



The main diagonal shows the alignment with itself.

Other lines show repetitive patterns within the sequence.

Biological Background

Evolutionarily related DNA or protein sequences have mutations:

- substitutions
- insertions or deletions.

When aligning sequences we can allow:

- mismatch (corresponding to substitution)
- gap insertion (corresponding to insertion or deletion)

The second one is called *indels*.

Measures for Sequence Similarities

Hamming distance: number of positions differ in the two strings.

Note: It is not to useful to compare DNA or protein sequences because it considers only substitution mutations.

Levenshtein distance: minimum number of editing operations needed to transform one sequence into the other, where the editing operations are insertion, deletion and substitution.

Note: A given editing sequence corresponds to a unique pairwise alignment, but the reverse is not true.

Example for Measures

s1 = TATAT

s2 = ATATA

Hamming distance = 5

Levenshtein distance = 2

(step 1: insert an ‘A’ in front of s1

step 2: delete the ‘T’ at the end of s1)

Intro to Dynamic Programming Solution

Construct a grid where characters of one sequence index the rows, and characters of the other index the columns.

Any path through the grid from the top left to the bottom right corner corresponds to an alignment.

Each segment in a path corresponds

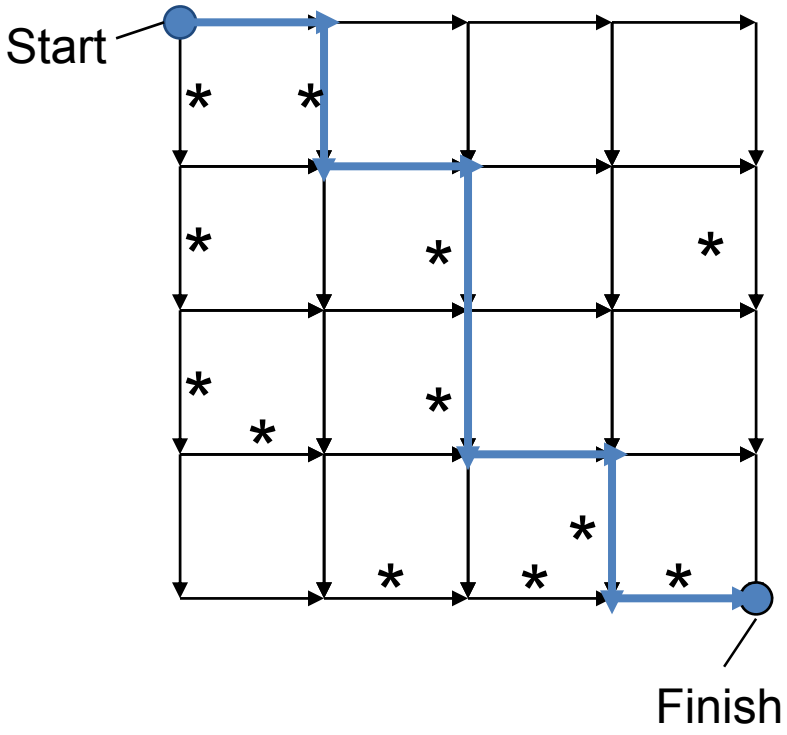
- an indel (if its direction is down or side-way)
- a match or a substitution (if its direction is diagonal).

We need to find the "optimal" path assuming that each segment has an associated cost.

Related problem: Manhattan Tourist Problem (MTP).

Manhattan Tourist Problem (MTP)

Find a path with the most number of attractions (*) in the Manhattan grid going from an upper West-side corner (Start) to a lower East-side corner (Finish) of Manhattan and traveling only eastward and southward.



MTP: Exhaustive (Brute Force) Solution

Generate *ALL* possible paths in the grid.

Output the best path as solution.

Guaranteed to find optimal solution.

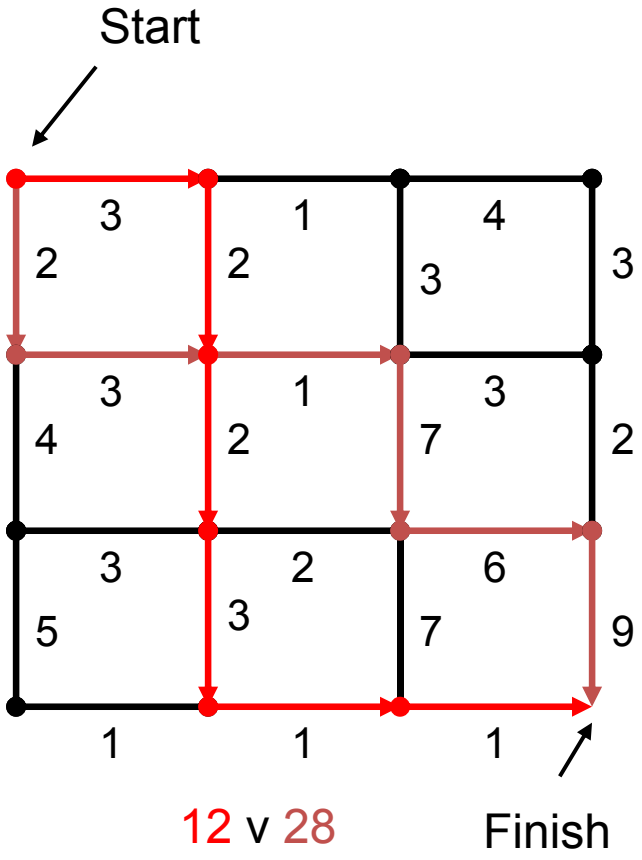
It is tractable if graph is not large.

Not feasible for even a moderately sized graph.

MTP: A Greedy Solution

At every vertex, choose the adjacent edge with the highest weight.

Easily achievable in polynomial time, but is unlikely to give the optimal solution, especially for larger graphs!



MTP: Dynamic Programming Solution

Store at each vertex the "value" of the optimal path ($s_{i,j}$) leading to that vertex.

Initialize $s_{0,0}$ to 0.

Now computing values in 1st row and 1st column ($s_{0,j}$ and $s_{i,0}$ for all i and j) is easy.

Finally we can compute the rest of the $s_{i,j}$ values as

$$s_{i,j} = \max \begin{cases} s_{i-1,j} + \text{weight of edge between } (i-1,j) \text{ and } (i,j) \\ s_{i,j-1} + \text{weight of edge between } (i,j-1) \text{ and } (i,j) \end{cases}$$

Generalized MTP

How to handle diagonal streets of Manhattan (like e.g. Broadway)?

The only difference is that each vertex has not two but three neighbors:

$$s_{i,j} = \max \begin{cases} s_{i-1,j} + \text{weight of edge between } (i-1,j) \text{ and } (i,j) \\ s_{i-1,j} + \text{weight of edge between } (i-1,j-1) \text{ and } (i,j) \\ s_{i,j-1} + \text{weight of edge between } (i,j-1) \text{ and } (i,j) \end{cases}$$

Travelling the Grid

The only additional issue is that one must decide on the order in which visit the vertices.

By the time a vertex is analyzed, the values for all its predecessors (neighbors) should be computed – otherwise we are in trouble.

The graph should be cycle free (DAG – Directed Acyclic Graph).

We need to traverse the vertices in a so-called topological order.

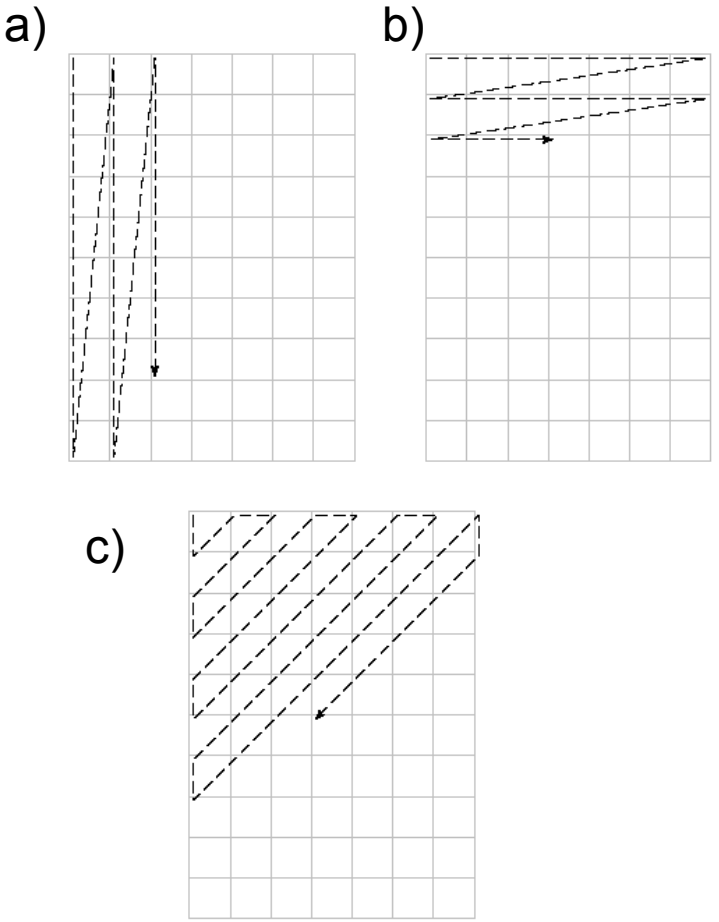
Topological Order for MTP

3 different strategies:

a) Column by column

b) Row by row

c) Along diagonals



Actual Optimal Route: Backtracking

The discussed algorithm computes the value of the optimal path leading to ‘Finish’.

However, we need to get the actual routing as well.

We can take up a second (trace-back) matrix and in each of it’s cells we store the neighbor that was used to get the max value for the associated vertex.

Then after finished computing the values, we can backtrack from cell (n, m) to cell (0, 0) of the trace-back matrix to recreate the optimal route.

Run Time Comparison

Exhaustive (brute force) solution

It take too long – $O(n) = 2^n$

Greedy solution

It is extremely fast – $O(n) = n$

Not acceptable because it usually misses the optimal solution.

Dynamic programming solution

It is fast – $O(n) = n^2$

It always finds an optimal solution.

Back to DP Solution of Alignment

We can apply the DP solution of MTP (using the alignment matrix).

We need to use a scoring mechanism for assigning value to each possible path.

Let us introduce a simple scoring schema:

$+1$: premium for matches (on diagonal edges)

$-\mu$: penalty for mismatch (on diagonal edges)

$-\sigma$: penalty for indel (on non-diagonal edges)

Value of a path:

$$match\# - \mu(\text{mismatch}\#) - \sigma\,(\text{indel}\#)$$

Scoring Matrix

Studies of mutations show that the different substitutions do not have the same frequency.

Therefore it is preferable to create a scoring matrix based on substitution probabilities and use the appropriate value from this matrix when computing a mismatch .

To generalize scoring, a (4+1) x(4+1) scoring matrix can be used. The addition of extra column/line is to handle indels (that is, to include the score for comparison of a gap character "-").

In the case of an amino acid sequence alignment, the scoring matrix would be of (20+1)x(20+1) size (PAM, BLOSUM).

PAM250 matrix (developed by Dayhoff)

	A	R	N	D	C	Q	E	G	H	I	L	K	M	F	P	S	T	W	Y	V
Ala (A)	2																			
Arg (R)	−2	6																		
Asn (N)	0	0	2																	
Asp (D)	0	−1	2	4																
Cys (C)	−2	−4	−4	−5	12															
Gln (Q)	0	1	1	2	−5	4														
Glu (E)	0	−1	1	3	−5	2	4													
Gly (G)	1	−3	0	1	−3	−1	0	5												
His (H)	−1	2	2	1	−3	3	1	−2	6											
Ile (I)	−1	−2	−2	−2	−2	−2	−2	−3	−2	−5										
Leu (L)	−2	−3	−3	−4	−6	−2	−3	−4	−2	2	6									
Lys (K)	−1	3	1	0	−5	1	0	−2	0	−2	−3	5								
Met (M)	−1	0	−2	−3	−5	−1	−2	−3	−2	2	4	0	6							
Phe (F)	−3	−4	−3	−6	−4	−5	−5	−5	−2	1	2	−5	0	9						
Pro (P)	1	0	0	−1	−3	0	−1	0	0	−2	−3	−1	−2	−5	6					
Ser (S)	1	0	1	0	0	−1	0	1	−1	−1	−3	0	−2	−3	1	2				
Thr (T)	1	−2	0	0	−2	−1	0	0	−1	0	−2	0	−1	−3	0	1	3			
Trp (W)	−6	2	−4	−7	−8	−5	−7	−7	−3	−5	−2	−3	−4	0	−6	−2	−5	17		
Tyr (Y)	−3	−4	−2	−4	0	−4	−4	−5	0	−1	−1	−4	−2	7	−5	−3	−3	0	10	
Val (V)	0	−2	−2	−2	−2	−2	−2	−1	−2	4	2	−2	2	−1	−1	−1	0	−6	−2	4

BLOSUM62 Matrix (developed by Henikoff)

	A	R	N	D	C	Q	E	G	H	I	L	K	M	F	P	S	T	W	Y	V
Ala (A)	4																			
Arg (R)	−1	5																		
Asn (N)	−2	0	6																	
Asp (D)	−2	−2	1	6																
Cys (C)	0	−3	−3	−3	9															
Gln (Q)	−1	1	0	0	−3	5														
Glu (E)	−1	0	0	2	−4	2	5													
Gly (G)	0	−2	0	−1	−3	−2	−2	6												
His (H)	−2	0	1	−1	−3	0	0	−2	8											
Ile (I)	−1	−3	−3	−3	−1	−3	−3	−4	−3	4										
Leu (L)	−1	−2	−3	−4	−1	−2	−3	−4	−3	2	4									
Lys (K)	−1	2	0	−1	−3	1	1	−2	−1	−3	−2	5								
Met (M)	−1	−1	−2	−3	−1	0	−2	−3	−2	1	2	−1	5							
Phe (F)	−2	−3	−3	−3	−2	−3	−3	−3	−1	0	0	−3	0	6						
Pro (P)	−1	−2	−2	−1	−3	−1	−1	−2	−2	−3	−3	−1	−2	−4	7					
Ser (S)	1	−1	1	0	−1	0	0	0	−1	−2	−2	0	−1	−2	−1	4				
Thr (T)	0	−1	0	−1	−1	−1	−1	−2	−2	−1	−1	−1	−1	−2	−1	1	5			
Trp (W)	−3	−3	−4	−4	−2	−2	−3	−2	−2	−3	−2	−3	−1	1	−4	−3	−2	11		
Tyr (Y)	−2	−2	−2	−3	−2	−1	−2	−3	2	−1	−1	−2	−1	3	−3	−2	−2	2	7	
Val (V)	0	−3	−3	−3	−1	−2	−2	−3	−3	3	1	−2	1	−1	−2	−2	0	−3	−1	4

Comparing PAM and BLOSUM

PAM matrices:

List the likelihood of change from one amino acid to another in homologous protein sequences and during evolution.

Based on a mutational model of evolution that assumes the changes occur according to a Markov process (each change at a site is independent of previous changes at that site)

BLOSUM matrices:

Based on an implicit evolutionary model and use the scores of local similarity of sections in the BLOCKS database

Affine Gap Penalties

In evolution a series of k indels is often the result of a single event rather than a series of k single mutation events.

Therefore using a fixed penalty σ for every elements of a series of consecutive indels is too severe.

More accurate to use a score for a gap of length x:

$$-(\rho + \sigma x)$$

where $\rho > 0$ is the penalty for introducing a gap (gap opening penalty) and

$\sigma > 0$ is the penalty for extending a gap (gap extension penalty)

Analyzing DP Solution

Advantages:

- It is guaranteed to find an optimal alignment given a particular scoring matrix.
- It is very fast when we need to compare only two sequences.

Disadvantage:

- In large-scale database searches in particularly since a large proportion of the sequences from the database will have essentially no significant match with the query sequence, and it would take intolerable amount of time.

More Issues with DP Solution

Several different alignments of two DNA or protein sequences may have the highest score.

Most sequence alignment programs provide only one optimal alignment.

In some cases additional alignments may have scores that are only somewhat lower than the optimal one.

These suboptimal alignments could sometimes be biologically more meaningful than the optimal one(s).

Some programs (e.g. LALIGN) can provide these additional alignments.

Global vs. Local Alignment

Global alignment

- Includes all of the sequences.
- Uses the DP algorithm as describe on previous slides.
- Each matrix position can have positive, negative or 0 scores.

Local alignment

- Includes only those parts of the sequences that provide a high-scoring alignment
- Uses the same DP with a modification: when a score gets negative at a matrix position, then the value is changed to 0 (terminating any alignment up to that point).

Word and k-tuple Methods

These are rapid methods that are used when dynamic programming is not fast enough.

They apply a heuristic approach and do not necessarily find the optimal alignment.

In the process of aligning two sequences they

- first search for identical short subsequences (so-called words or k-tuples)

- and then join these words into an alignment using dynamic programming method.

The algorithms FASTA and BLAST are based on this approach and their detailed discussion is in Chapter 4.

References to DP Solution

Global alignment

- Needleman-Wunsch algorithm
  - Needleman, Wunsch, 1970. A general method applicable to the search for similarities in the amino acid sequence of two proteins. J. Mol. Biol. 48: 443-53

Local alignment

- Smith-Waterman algorithm
  - Smith, Waterman, 1981. Identification of common molecular subsequences. J. Mol. Biol. 147: 195-97

Multiple Sequence Alignment

Comparing multiple sequences and trying to discover similarities between them.

A faint similarity between two sequences becomes significant if present in many.

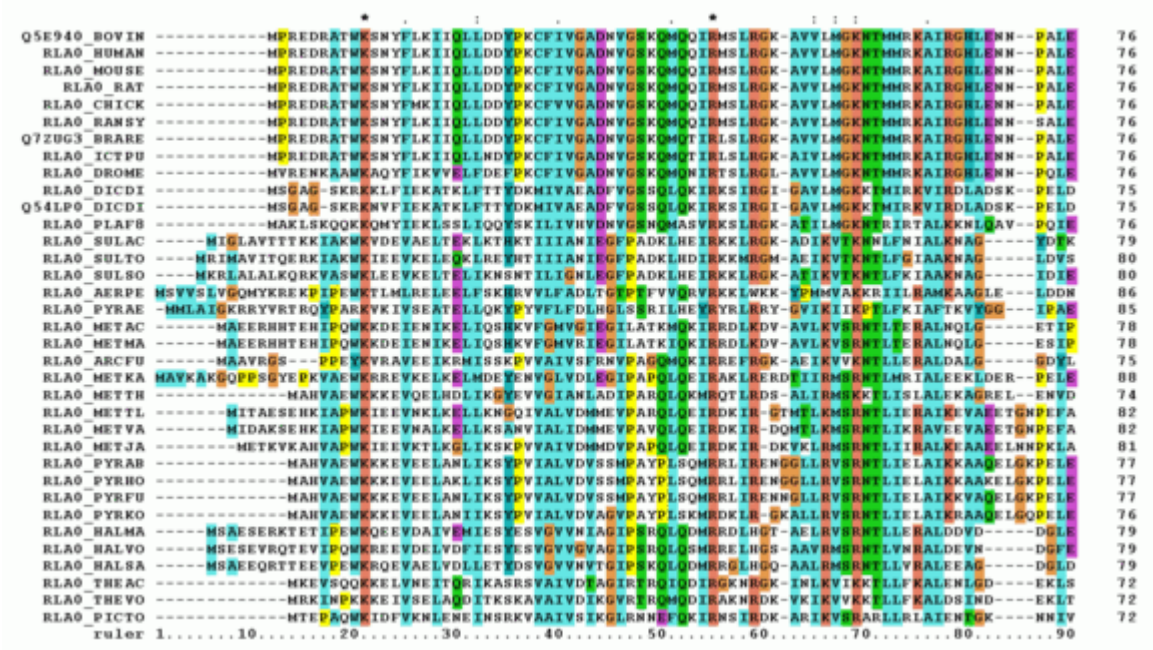
Multiple alignments can reveal subtle similarities that pairwise alignments do not reveal.

Multiple alignments are also used to aid in establishing evolutionary relationships by constructing phylogenetic trees.

It can also be useful in genome sequencing.

Visualization of Multiple Sequence Alignment

Visualization by software tools can illustrate mutation such as point mutations (appearing as differing characters) and insertion/deletion mutations (indels, appearing as hyphens).



First 90 positions of a protein multiple sequence alignment from several organisms, generated with ClustalX (Windows interface for a ClustalW multiple sequence alignment)

Types of Multiple Alignment

Just as at pairwise alignments, we could have

- Global alignment – attempts to align the entire sequences that participates in the process
- Local alignment – looks for well conserved regions in the sequences

Relationship Between Pairwise and Multiple Sequence Alignments

From an optimal multiple alignment, we can infer pairwise alignments between every pairs of sequences, but they are not necessarily the optimal alignments.

We have even more difficulties with the reverse problem; in some cases pairwise alignments cannot be combined into multiple alignments.

Scoring of Multiple Sequence Alignments

There are different ways to evaluate (score) multiple sequence alignments:

- number of exact matches (only those columns count that have the same character in each sequence; it has limited value – useful only for very similar sequences)
- entropy score (see details on next slide)
- sum of pairs (SP, the sum of the scores of all possible pairwise alignments)

Entropy Score

Determine the frequencies of occurrence of each letter in each column of the sequences.

Compute entropy of each column:

$$-\sum_{X=A,T,G,C} p_X \log p_X$$

Entropy for a multiple alignment is the sum of entropies of its columns:

$$\Sigma_{\text{over all columns}} \Sigma_{X=A,T,G,C} p_X \log p_X$$

Methods for Multiple Alignment

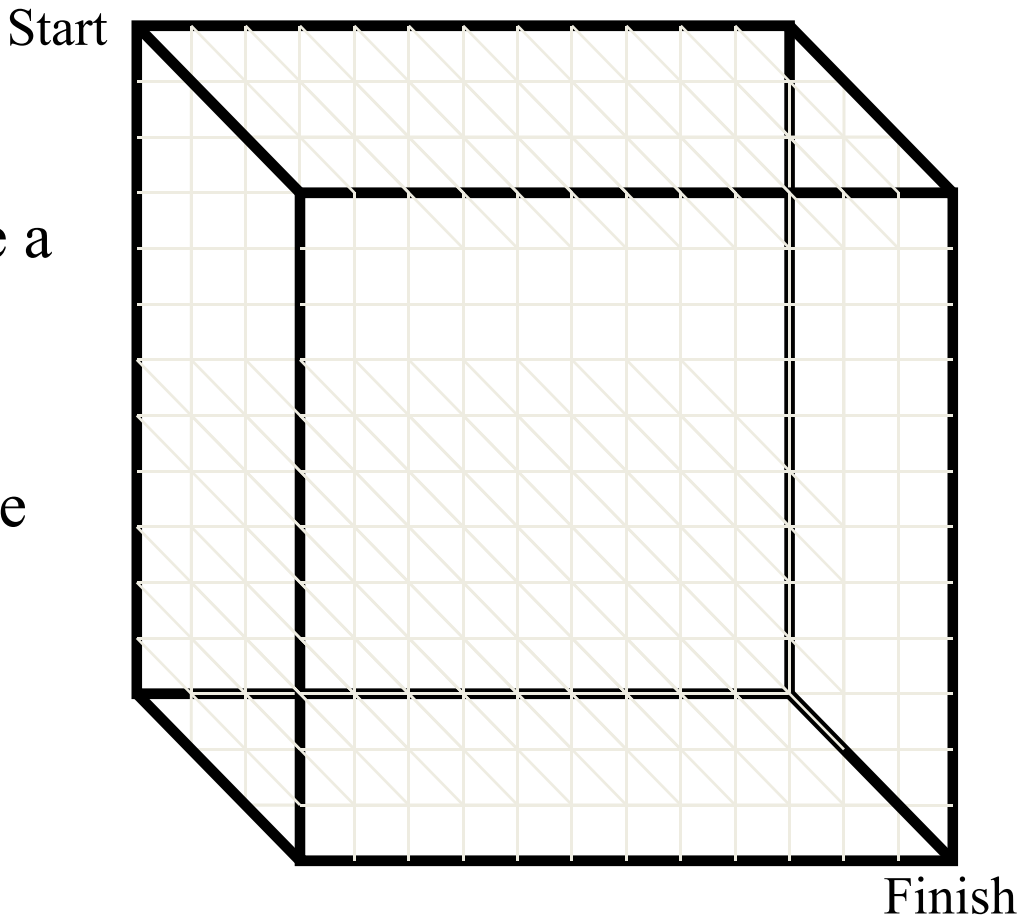
- Extending the pairwise sequence alignment
- Progressive alignment of the sequences
- Iterative methods
- Genetic algorithm
- Hidden Markov Models (HMM)

Note: Multiple sequence alignment algorithms are computationally difficult to produce and most real-life problems are NP-complete and therefore heuristics are used.

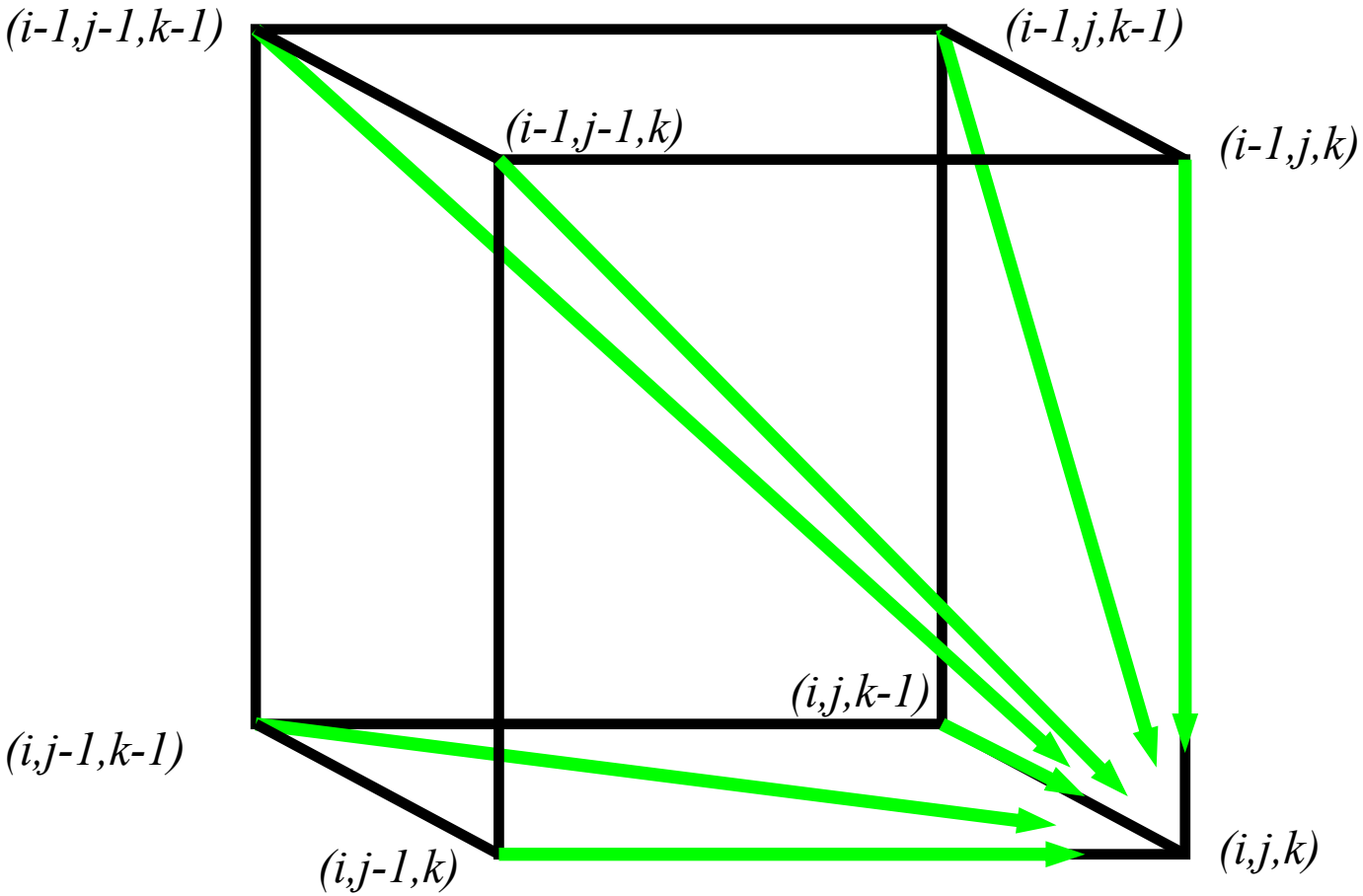
Extending Pairwise Alignment

For 3 sequences it is easy: use a 3-D “Manhattan Cube”, with each axis a sequence to align.

For global alignments, find the optimal path from Start to Finish.



Architecture of the 3-D alignment



Algorithm for Extending Pairwise Alignment

- For each vertex it computes the maximum value considering **all** neighbors (predecessors):

$$s_x = \text{max of } s_y + \text{weight of vertex (y, x) where}$$

$$y \in \text{Predecessors}(x)$$

- There are 7 neighbors for 3 sequences, and generally 2^k-1 neighbors for ***k*** sequences.
- A ***k***-dimensional scoring matrix is needed for ***k*** sequences.

Run Time for Extending Pairwise Alignment

- For three sequences of length $\boldsymbol{n}$, the run time is quite acceptable $7\boldsymbol{n}^3$; $O(\boldsymbol{n}^3)$.
- For $\boldsymbol{k}$ sequences, if we use a $\boldsymbol{k}$-dimensional Manhattan, the run time is $(2^{\boldsymbol{k}}-1)(\boldsymbol{n}^{\boldsymbol{k}})$; $O(2^{\boldsymbol{k}}\boldsymbol{n}^{\boldsymbol{k}})$.
- Thus extending the pairwise sequence alignment for larger number of sequences is impractical since the running time is exponentially grows.
- Therefore it is rarely used for more than three or four sequences.

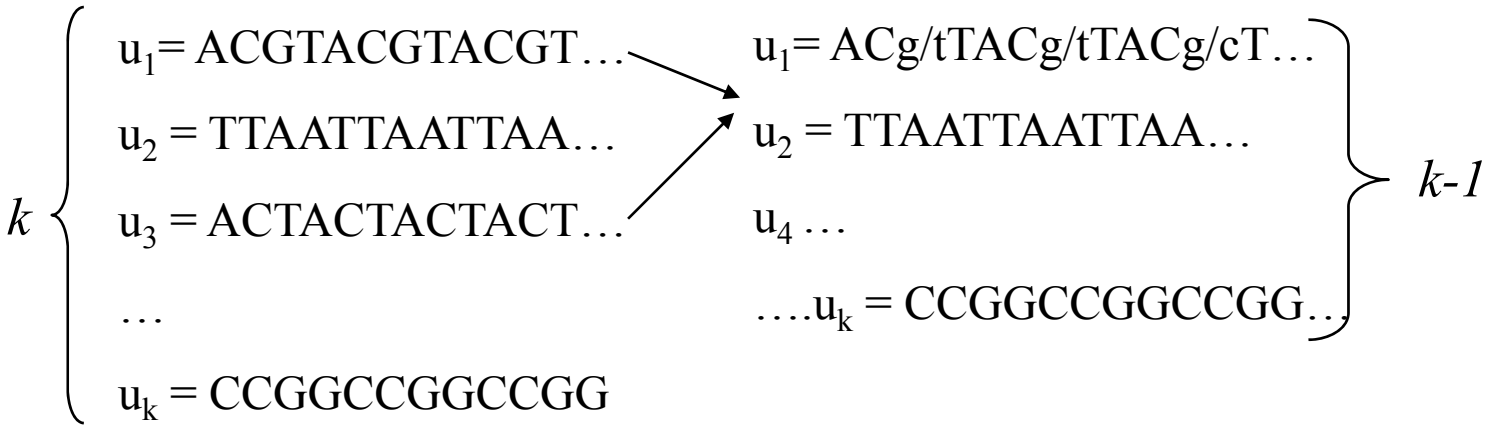
Progressive Alignment of Sequences 1.

Greedy approach:

- Select (with pairwise alignment) the pair of sequences with the highest similarity value (as seed)
- Merge them together into a so-called profile and replace them with the resulting sequence
- Repeat the process on the reduced multiple alignment of k-1 sequences

Note: It may go off-track by choosing a spuriously strong pairwise alignment (that is, a bad seed).

Example for Greedy Approach



Sequences u_1 and u_3 are combined into a profile and replaced.

Progressive Alignment of Sequences 2.

Improved approach: CLUSTALW

- Performs pairwise alignments on all possible pairs of the sequences (this could use a rapid k-tuple solution like FASTA).

- Based on the alignment scores it produces a phylogenetic tree using the so-called neighbor-joining method.

- Aligns the sequences using the pairwise dynamic programming algorithm, guided by the phylogenetic relationships indicated by the tree, inserting gaps as necessary.

Problems with Progressive Alignment

The major problem is that the final resulting multi-alignment heavily depends on the choice of the initial pairwise alignment (that is, errors of initial choice will propagate the result). This problem is more serious when the initial choice is between more distantly related sequences.

Choice of suitable scoring matrix and gap penalties affect the result.

Previous alignment information is lost when sequences are merged into profiles.

Iterative Methods for Multiple Alignment

This method attempts to correct these problems:

- Repeatedly realigns subgroups of sequences and then aligns these subgroups into a global alignment.
- Continues the iteration while the sum of the alignment scores for each pair of sequences ("overall score", SP) in the multiple alignment can be improved.
- Number of such programs exist (MultiAlin, PRRP, DIALIGN).

Genetic Algorithms

They are general type of data mining algorithms.

Main representative is SAGA (Sequence Alignment by Genetic Algorithm):

- Creates an initial (random) set of 100 multi segment alignments (msa) as G_0
- Selects some msa-s ("parents") that best fit to generate offspring msa-s for next generation (G_{k+1})
- Evaluate the fitness of the population of G_{k+1} (using an objective function, a measure of multiple alignment quality)
- If population is not stabilized then stop else generate next G.

Comments on SAGA

During "breeding" (creation of next generation) typically 50% of the fittest individuals from the previous generation are kept and the rest is replaced with the generated offspring sequences to form the new generation.

As stabilization criteria, SAGA checks if unable to make improvement for some specified number of generations (typically 100).

There is no valid proof that the optimum can be reached, even in an infinite amount of time.

SAGA is fairly slow for large test cases (with >20 or so sequences)

Hidden Markov Models

It is a probabilistic model that assigns likelihoods to possible combinations of gaps, matches, and mismatches and determines the most likely MSA or set of possible MSAs:

- It is initiated with a directed acyclic graph (DAG) known as a partial-order graph, which consists of a series of nodes representing possible entries in the columns and the estimates of transition probabilities.
- Sequences to be aligned are used as training data set and the DAG (representation of HMM) is readjusted accordingly.
- The trained model provides the most likely path for each sequence and thus the msa for the entire set of sequences.

Pros and Cons for HMM

Advantages

- Offer significant improvements in computational speed especially for sequences with overlapping subsequences.
- Has strong foundation in probability theory
- No sequence ordering is needed.
- Guesses of gap penalties are not needed.
- Can produce the highest-scoring output (msa), but can also provide a set of possible alignments that can then be evaluated for biological significance.
- Can be used for both global and local alignments.

Pros and Cons for HMM (continued)

Advantages

- Can be used for both global and local alignments.
- Experimentally derived information can also be used.

Disadvantages

- At least 20 sequences (and in some special cases many more) are needed for training purpose.
- The success of applying HMM significantly depends on providing an appropriate initial model (e.g. should properly capture the expected amino acid frequencies in proteins).

Multiple Sequence Alignment Programs

- ClustalW

  Higgins, Thompson, Gibson, 1996. Using Clustal for multiple sequence alignment.

  Methods Enzymol. 366:383-402

  http://www.clustal.org/

- SAGA

  Notredame, Higgins, 1996. Sequence Alignment by Genetic Algorithm

  Nucleic Acid Research, 24:1515-24

  www.tcoffee.org/Projects_home_page/saga_home_page.html

Multiple Sequence Alignment Programs

(continued)

- Sequence Alignment and Modeling Software (SAM)

  Krogh et al., 1994. Hidden Markov models in computational biology. J. Mol. Biol. 235:1501-31

  http://compbio.soe.ucsc.edu/sam.html

- HMMER

  Eddy, 1998. Profile hidden Markov models. Bioinformatics 14: 755-63

  http://hmmer.janelia.org/

Problems with Multiple Alignment

Multidomain proteins evolve not only through point mutations but also through domain duplications and domain recombination.

Although multiple sequence alignment is a 30 year old problem, there were no multiple sequence alignment approaches for aligning rearranged sequences (i.e., multi-domain proteins with shuffled domains) prior to 2002.

Often impossible to align all protein sequences throughout their entire length.

History of Multiple Sequence Alignment

1975 Sankoff

Formulated multiple alignment problem and gave dynamic programming solution

1988 Carrillo-Lipman

Branch and Bound approach for MSA

1990 Feng-Doolittle

Progressive alignment

1994 Thompson-Higgins-Gibson-ClustalW

Most popular multiple alignment program

1998 Morgenstern et al.-DIALIGN

Segment-based multiple alignment

2000 Notredame-Higgins-Heringa-T-coffee

Using the library of pairwise alignments

2004 MUSCLE