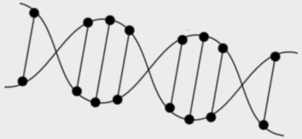
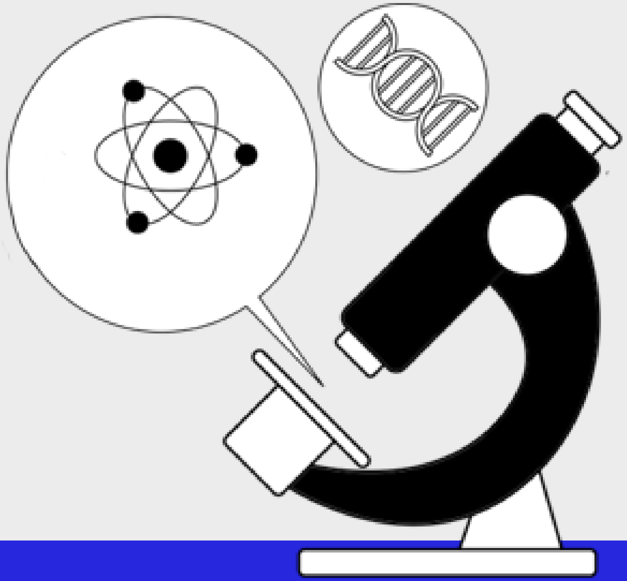


Multiple Alignment

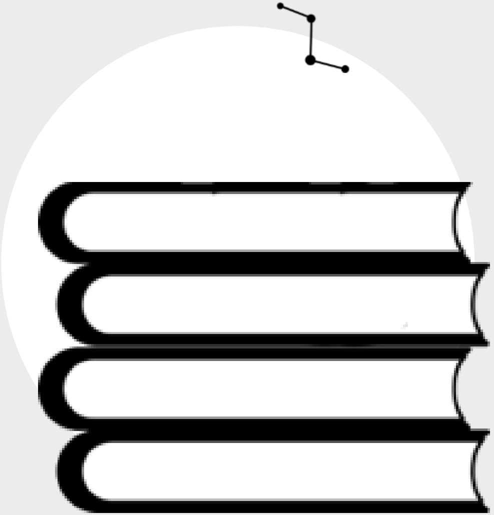
Generalized DP

Saeedeh Akbari
Department of Computer Engineering
Sharif University of Technology
Fall 2023



Adapted with modifications from lecture notes prepared by Phillip Compeau
Bioinformatics Algorithms: An Active Learning Approach

Table of contents



01

Introduction

02

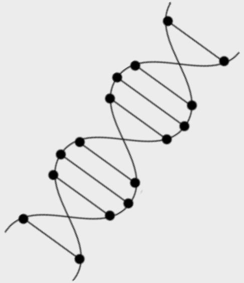
Solve the Problem

03

Scoring MSA

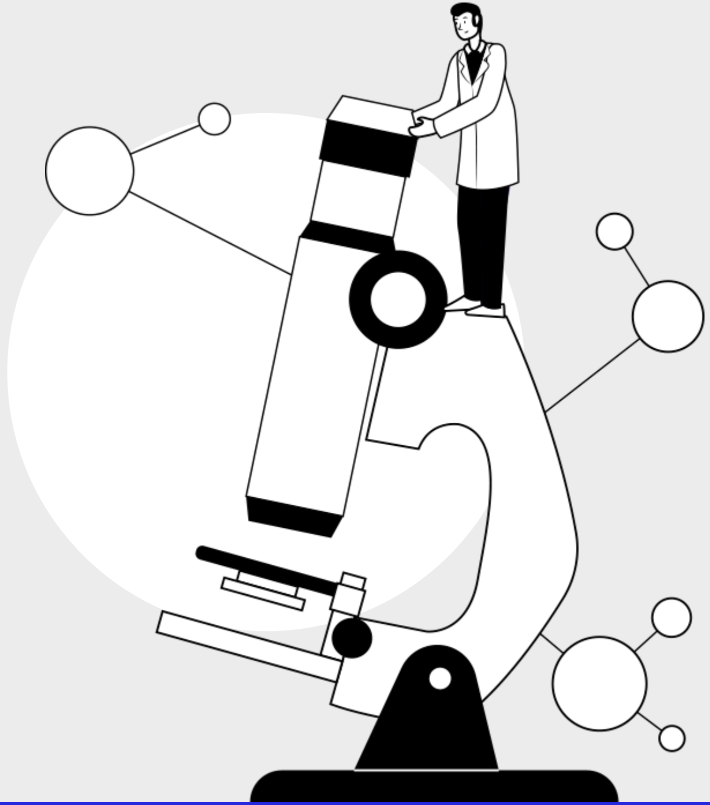
04

Other Methods



01

Introduction



Why MSA

- If sequence similarity is weak, pairwise alignment may not identify biologically related sequences.

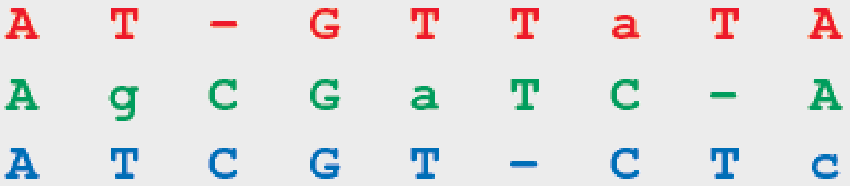
- Simultaneous comparison of many sequences often allows us to find similarities that pairwise sequence comparison fails to reveal.

- Bioinformaticians sometimes say that while pairwise alignment whispers, multiple alignment shouts.

From pairwise to multiple alignment

- Alignment of 2 sequences is represented as a 2-row matrix

- In a similar way, we represent alignment of 3 sequences as a 3-row matrix



```
A T - G T T a T A
A g C G a T C - A
A T C G T - C T c
```

- Score: more conserved columns, better alignment

What is MSA

- A model
- Indicates relationship between residues of different sequences
- Reveals similarity/dissimilarity.

Multiple Alignment Problem: *Find the highest scoring alignment between multiple strings.*

- **Input**: A collection of t strings
- **Output**: A multiple alignment of these strings having maximal score.

MSA Applications

MSA is central to many bioinformatics applications:

- Phylogenetic tree
- Motifs
- Patterns
- Structure prediction (RNA, protein)

Multiple alignment: History

1975 Sankoff

    Formulated multiple alignment problem and gave DP 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 DIALIGN (Segment-based multiple alignment)

2000 T-coffee (consensus-based)

2004 MUSCLE

2005 ProbCons (uses Bayesian consistency)

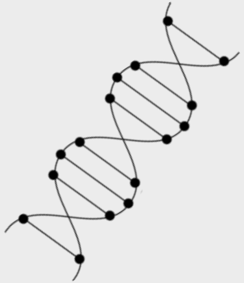
2006 M-Coffee (consensus meta-approach)

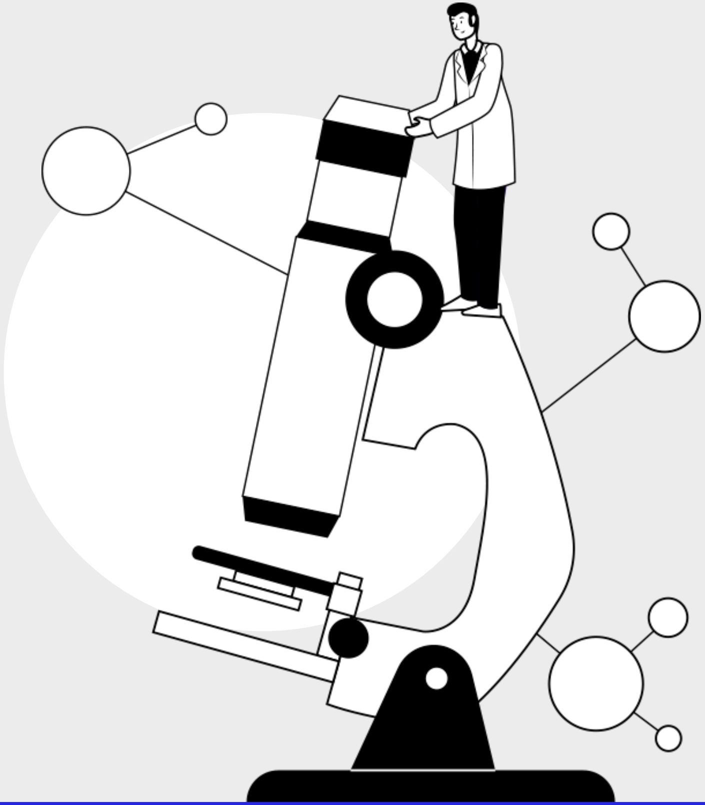
2006 Expresso (3D-Coffee; use structural template)

2007 PROMALS (profile-profile alignment)

02

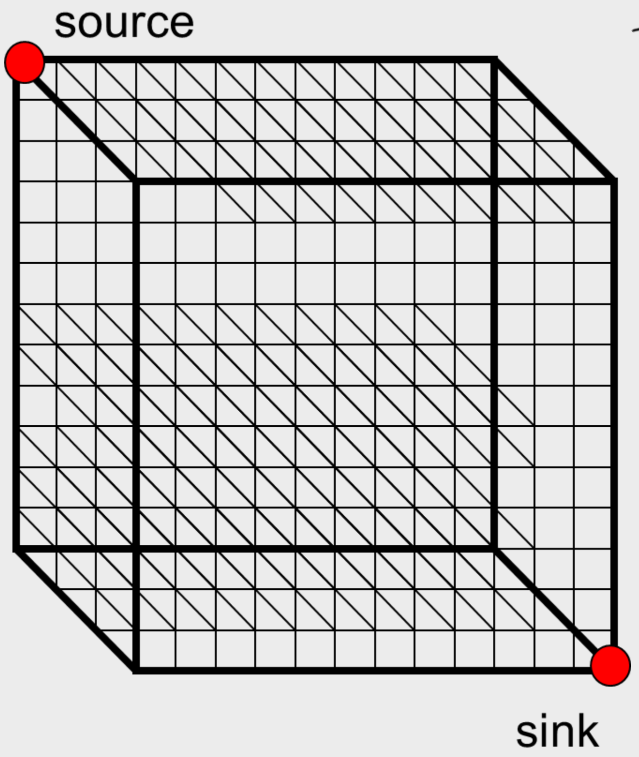
Solve the Problem



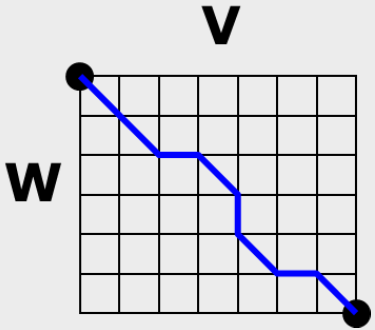


Aligning three sequences

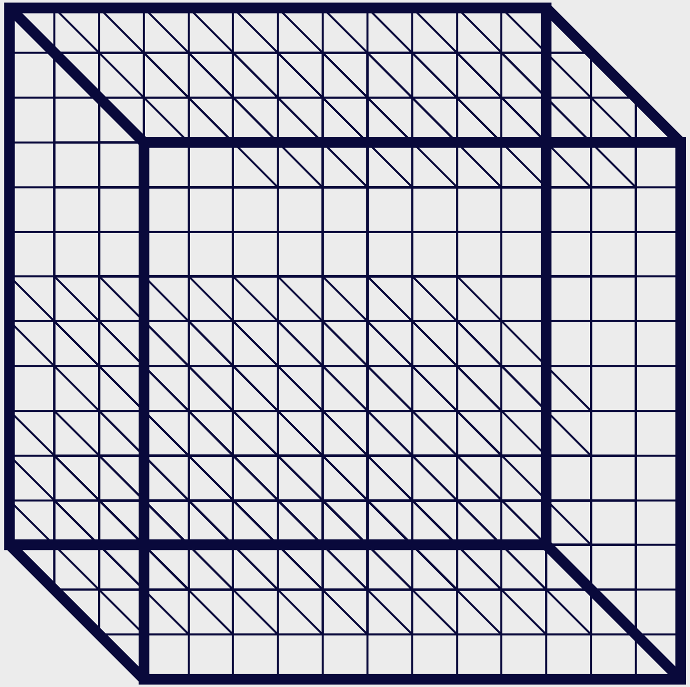
- Same strategy as aligning two sequences
- Use a 3-D “Manhattan Cube”, with each axis representing a sequence to align
- For global alignments, go from source to sink



2D vs 3D alignment grid

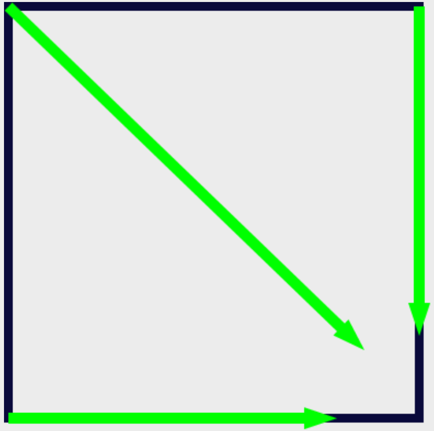


2D table

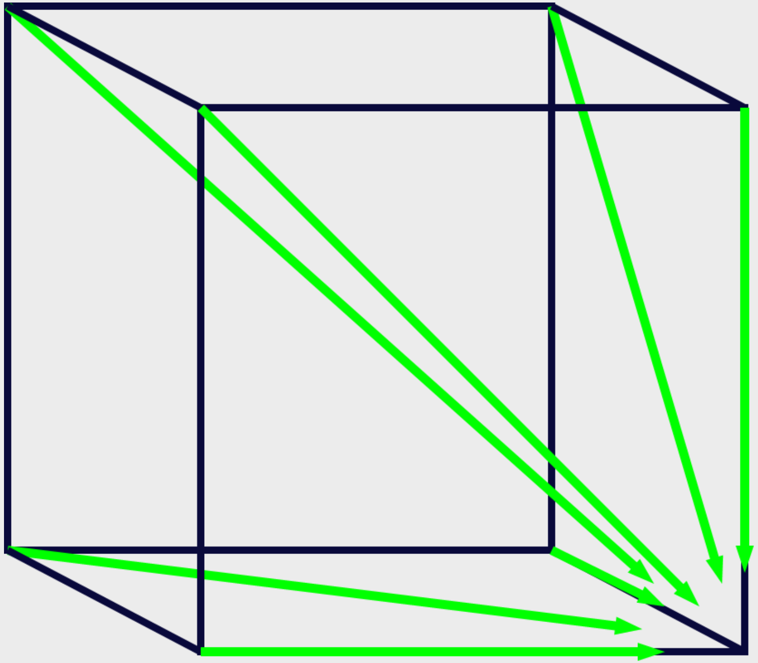


3D graph

DP recursion (3 edges vs 7)

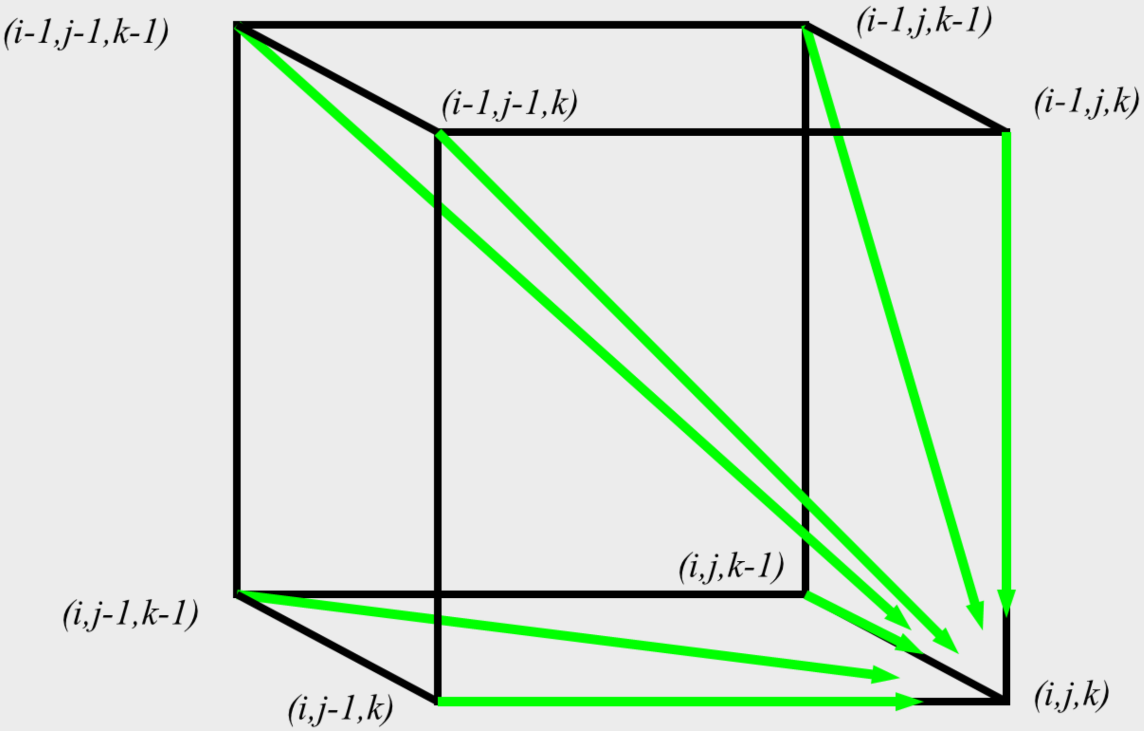


Pairwise: 3 possible paths (match/mismatch, insertion, and deletion)



In **3D**, 7 edges in each unit cube

Architecture of 3D alignment cell



Multiple alignment (dynamic programming)

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

cube diagonal: no indels

face diagonal: one indel

edge diagonal: two indels

- $\delta(x, y, z)$ is an entry in the 3D scoring matrix

MSA: running time

- For 3 sequences of length n, the run time is $7n^3$; O(n^3)
- For k sequences, build a k-dimensional Manhattan, with run time (2^k-1)(n^k); O(n^k)
- Conclusion: dynamic programming approach for alignment between two sequences is easily extended to k sequences (simultaneous approach) but it is impractical due to exponential running time.
- Computing exact MSA is computationally almost impossible, and in practice heuristics are used (progressive alignment)

Greedy MSA Algorithm

1. Starts by selecting the two strings having the highest scoring pairwise alignment (among all possible pairs of strings)

2. Uses this pairwise alignment as a building block for iteratively adding one string at a time to the growing multiple alignment.

3. Select the string having maximum score against the current alignment at each stage.

➔ Problem of constructing a multiple alignment of *t* sequences is reduced to constructing *t* alignments

Profile representation of multiple alignment

Alignment

T	C	G	G	G	–	g	T	T	T	t	t
c	C	–	–	t	G	A	c	T	T	a	C
a	C	G	–	G	G	A	T	T	T	t	C
T	t	G	G	G	–	A	c	T	T	t	t
a	–	–	–	G	–	–	–	T	–	C	–
T	t	G	G	G	G	A	c	T	T	C	C
T	C	G	–	–	G	A	T	T	c	a	t
–	–	–	G	G	G	A	T	T	c	C	–
T	a	G	G	G	G	A	a	c	–	–	C
T	C	G	G	G	t	A	T	a	a	C	C

Profile

A:	.2	.1	0	0	0	0	.8	.1	.1	.1	.2	0
C:	.1	.5	0	0	0	0	0	.3	.1	.2	.4	.5
G:	0	0	.7	.6	.8	.6	.1	0	0	0	0	0
T:	.6	.2	0	0	.1	.1	0	.5	.8	.6	.2	.3

Aligning alignments/profiles

Given two alignments, can we align them?

```
x GGGCACTGCAT
y GGTTACGTC--    Alignment 1
z GGGAACTGCAG

w GGACGTACC--    Alignment 2
v GGACCT-----
```

Aligning alignments/profiles

Given two alignments, can we align them?
Hint: use alignment of corresponding profiles

```
x  GGGCACTGCAT
y  GGTTACGTC--    Combined Alignment
z  GGGAACTGCAG
w  GGACGTACC--
v  GGACCT-----
```

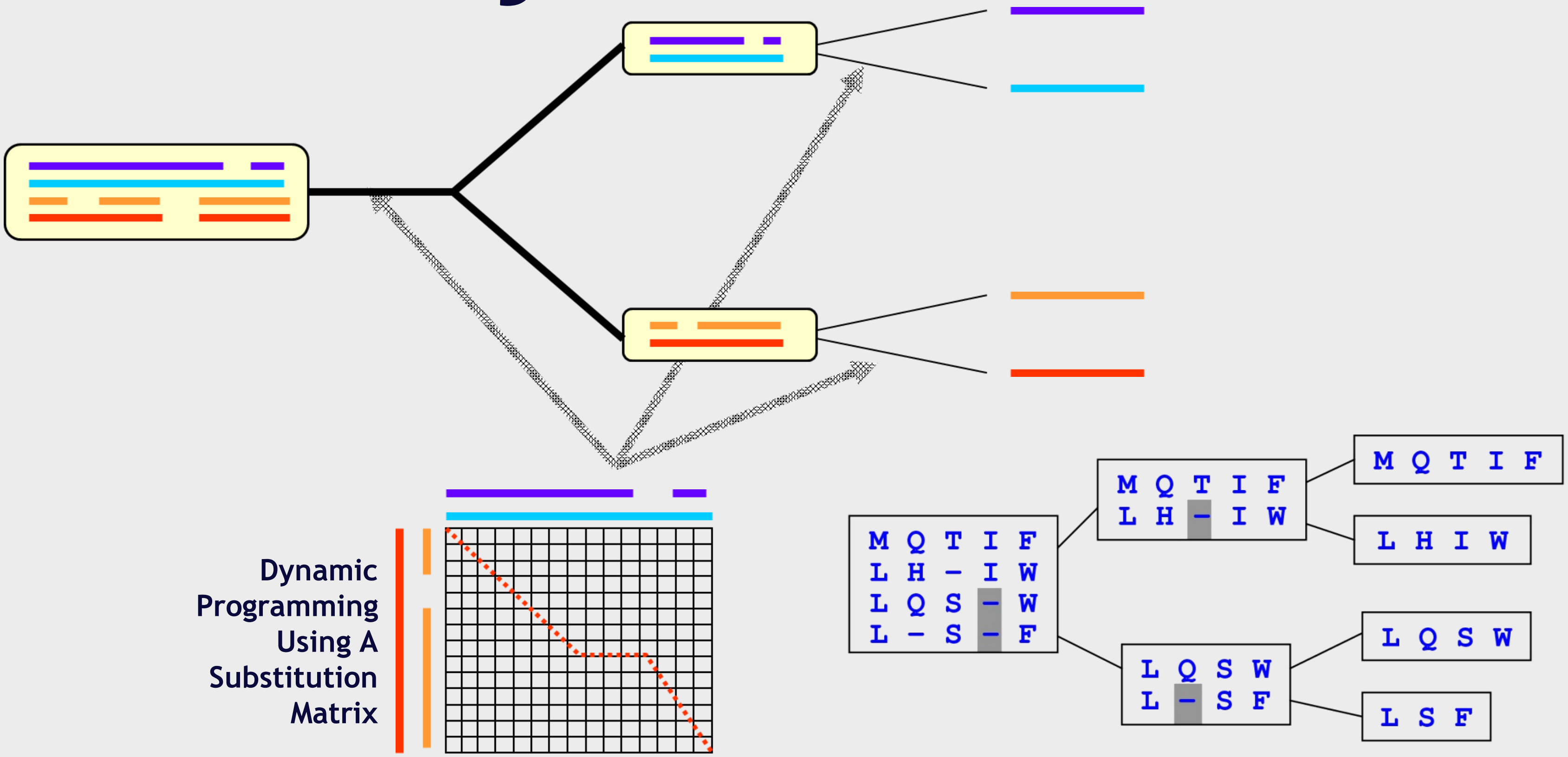
Progressive alignment

- Progressive alignment uses guide tree
- Sequence weighting & scoring scheme and gap penalties
- Progressive alignment works well for close sequences, but deteriorates for distant sequences
  - Gaps in consensus string are permanent
  - Use profiles to compare sequences

ClustalW

- Popular multiple alignment tool today
- ‘W’ stands for ‘weighted’ (sequences are weighted differently).
- Three-step process
  1. Construct pairwise alignments
  2. Build guide tree
  3. Progressive alignment guided by the tree

ClustalW algorithm



Dynamic Programming Using A Substitution Matrix

Step 1: Pairwise alignment

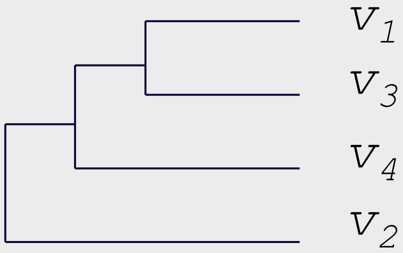
- Aligns each sequence again each other giving a similarity matrix
- Similarity = exact matches / sequence length (percent identity)

	$\mathbf{v}_1$	$\mathbf{v}_2$	$\mathbf{v}_3$	$\mathbf{v}_4$
$\mathbf{v}_1$	–			
$\mathbf{v}_2$	.17	–		
$\mathbf{v}_3$	.87	.28	–	
$\mathbf{v}_4$	.59	.33	.62	–

(.17 means 17 % identical)

Step 2: Guide tree

	v_1	v_2	v_3	v_4
v_1	–			
v_2	.17	–		
v_3	.87	.28	–	
v_4	.59	.33	.62	–



Calculate:

$v_{1,3}$ = alignment (v_1, v_3)

$v_{1,3,4}$ = alignment$((v_{1,3}),v_4)$

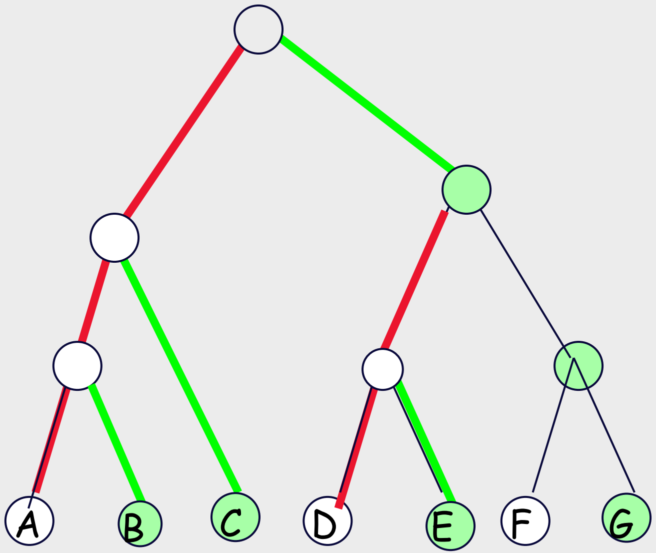
$v_{1,2,3,4}$ = alignment$((v_{1,3,4}),v_2)$

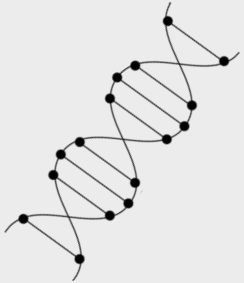
ClustalW uses NJ to build guide tree;

Guide tree *roughly* reflects evolutionary relations

Step 3: Tree based recursion

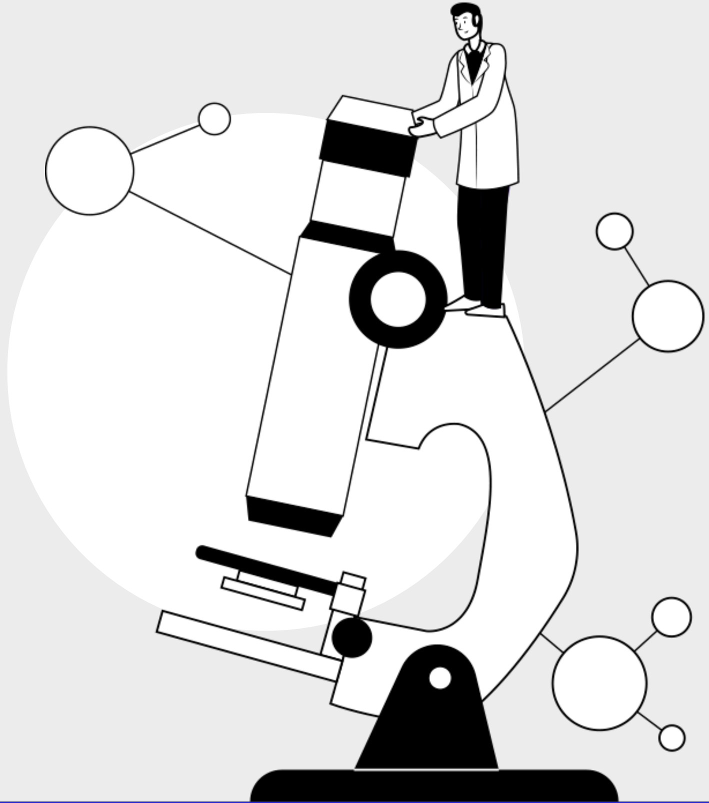
```
Align ( Node N)
        {
        if ( N->left_child is a Node)
                A1=Align ( N->left_child)

        else if ( N->left_child is a Sequence)
                A1=N->left_child

        if (N->right_child is a node)
                A2=Align (N->right_child)

        else if ( N->right_child is a Sequence)
                A2=N->right_child

        Return dp_alignment (A1, A2)
        }
```





03

Scoring MSA



Progressive alignment: Scoring scheme

- Scoring scheme is arguably the most influential component of the progressive algorithm
- Matrix-based algorithms
  - ClustalW, MUSCLE, Kalign
  - Use a substitution matrix to assess the cost of matching two symbols or two profiled columns
  - Once a gap, always a gap
- Consistency-based schemes
  - T-Coffee, Dialign
  - Compile a collection of pairwise global and local alignments (primary library) and to use this collection as a position-specific substitution matrix

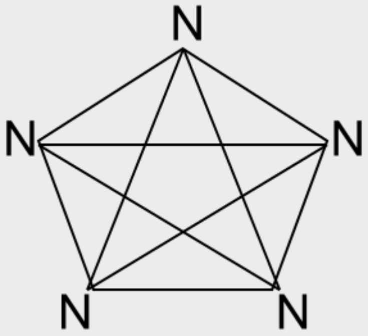
Substitution matrix based scoring

- Sum of pairs (SP score)
- Tree based scoring
- Entropy score

Sum of pairs score (SP score)

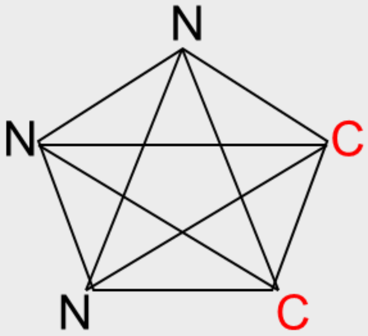
```
Seq   Column-A -B
1          ......N...............N......
2          ......N...............N......
3          ......N...............N......
4          ......N...............C......
5          ......N...............C......
```

(BLOSUM62)



Score= 10 * S(N,N)

= 10 * 6 = 60



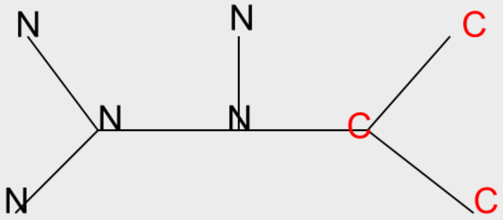
Score= 3 * S(N,N) + 6 * S(N,C) + S(C,C)

= 3 * 6 + 6 * (-3) + 9 = 9

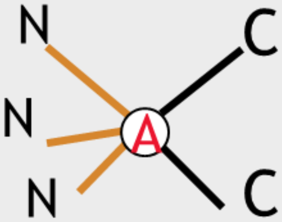
Problem: over-estimation of the mutation costs
(assuming each sequence is the ancestor of itself; requires a weighting scheme)

Tree-based scoring

```
Seq   Column-A -B
1          ......N..............N......
2          ......N..............N......
3          ......N..............N......
4          ......N..............C......
5          ......N..............C......
```



“Real” tree:
Cost = 1
But we do not know the tree!



Star tree
Cost=2
But the tree is wrong!

Entropy-based scoring

In information theory, entropy is a measure of the uncertainty associated with a random variable (a ,means to quantify information using some kind of currency, usually bits. The rarer, or equivalently more interesting, a thing is, the more bits its worth). The entropy H of a discrete random variable X with possible value x_1, …, x_2 is H(X) = E(I(X)), where I(X) is the information content of X.

If p denotes the probability mass function of X then the entropy is:

$$H(X) = \sum_i p(x_i)I(x_i) = -\sum_i p(x_i)log_2p(x_i)$$

Assume a genome has the following frequencies in its DNA:

$$P(A) = 0.2, p(T) = 0.2, p(C) = 0.3, p(G) = 0.3$$

Then its entropy is:

$$-(0.2log_2(0.2) + 0.2log_2(0.2) + 0.3log_2(0.3) + 0.3log_2(0.3)) = 1.97.$$

Entropy: Example

$$entropy\begin{pmatrix} A \\ A \\ A \\ A \end{pmatrix} = 0$$

$$entropy\begin{pmatrix} A \\ T \\ G \\ C \end{pmatrix} = -\sum \frac{1}{4}\log\frac{1}{4} = -4(\frac{1}{4} * -2) = 2$$

Given a DNA sequence, what is its maximum entropy?

Alignment entropy

- Define frequencies for the occurrence of each letter in each column of multiple alignment

  $p_A = 1$, $p_T=p_G=p_C=0$ (1st column)

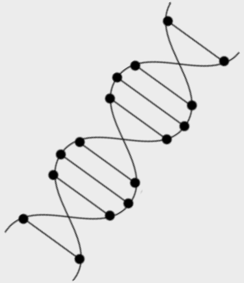
  $p_A = 0.25$, $p_T = p_G= 0$, $p_C= 0.75$ (2nd column)

  $p_A = 0.25$, $p_T = 0.25$, $p_C=0.25$ $p_G=0.25$ (3rd column)

- Compute entropy of each column

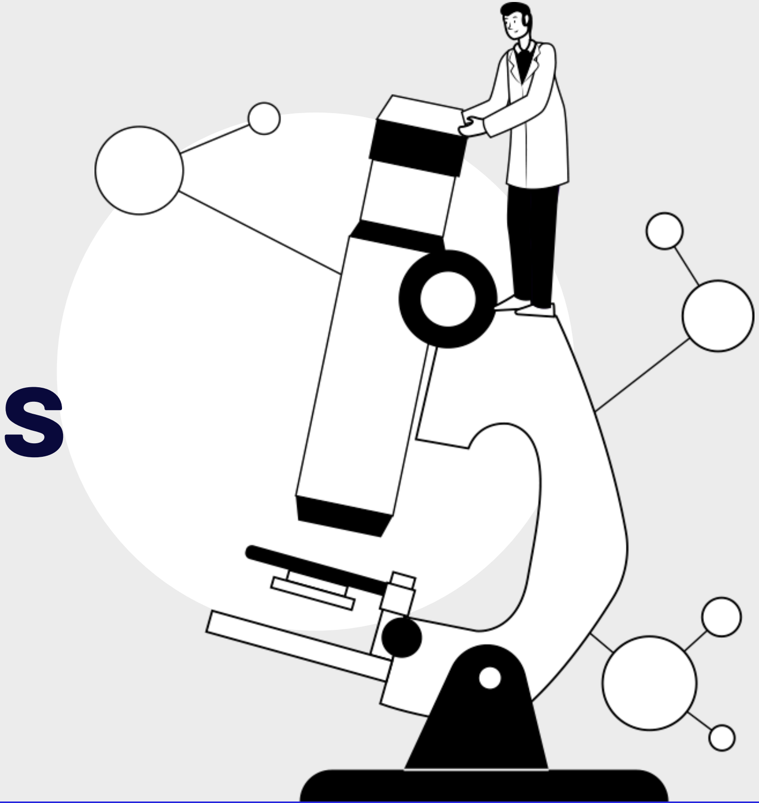
A	A	A
A	C	C
A	C	G
A	C	T
0	**0.811**	**2.0**

Alignment entropy= 2.811



04

Other Methods



Consistency-based approaches

- T-Coffee
    - M-Coffee & 3D-Coffee (Expresso)
- Principle
    - Primary library
    - Library extension

T-Coffee: Primary library

Input sequences

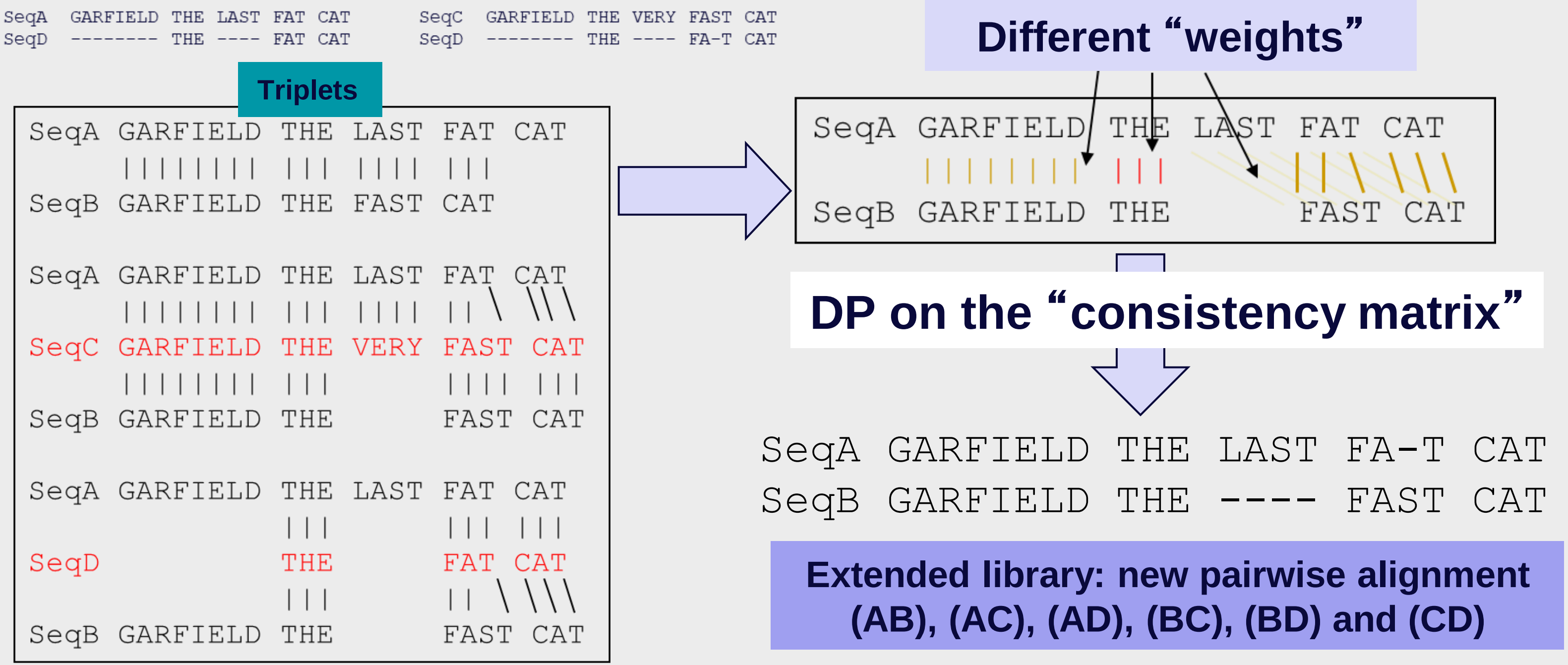
```
SeqA  GARFIELD THE LAST FAT CAT
SeqB  GARFIELD THE FAST CAT
SeqC  GARFIELD THE VERY FAST CAT
SeqD  THE FAT CAT
```

Primary library: collection of global/local pairwise alignments

```
SeqA  GARFIELD THE LAST FAT CAT    SeqB  GARFIELD THE ---- FAST CAT
SeqB  GARFIELD THE FAST CAT        SeqC  GARFIELD THE VERY FAST CAT

SeqA  GARFIELD THE LAST FA-T CAT   SeqB  GARFIELD THE FAST CAT
SeqC  GARFIELD THE VERY FAST CAT   SeqD  -------- THE FA-T CAT

SeqA  GARFIELD THE LAST FAT CAT    SeqC  GARFIELD THE VERY FAST CAT
SeqD  -------- THE ---- FAT CAT    SeqD  -------- THE ---- FA-T CAT
```

T-Coffee: Library extension

```
SeqA  GARFIELD THE LAST FAT CAT      SeqB  GARFIELD THE ---- FAST CAT
SeqB  GARFIELD THE FAST CAT          SeqC  GARFIELD THE VERY FAST CAT

SeqA  GARFIELD THE LAST FA-T CAT     SeqB  GARFIELD THE FAST CAT
SeqC  GARFIELD THE VERY FAST CAT     SeqD  -------- THE FA-T CAT

SeqA  GARFIELD THE LAST FAT CAT      SeqC  GARFIELD THE VERY FAST CAT
SeqD  -------- THE ---- FAT CAT      SeqD  -------- THE ---- FA-T CAT
```



Triplets

```
SeqA GARFIELD THE LAST FAT CAT
     |||||||| ||| |||| |||
SeqB GARFIELD THE FAST CAT

SeqA GARFIELD THE LAST FAT CAT
     |||||||| ||| |||| || \ \\\
SeqC GARFIELD THE VERY FAST CAT
     |||||||| |||      |||| |||
SeqB GARFIELD THE      FAST CAT

SeqA GARFIELD THE LAST FAT CAT
              |||      ||| |||
SeqD          THE      FAT CAT
              |||      || \ \\\
SeqB GARFIELD THE      FAST CAT
```

Different “weights”

```
SeqA GARFIELD THE LAST FAT CAT
     |||||||| |||      ||\ \\\
SeqB GARFIELD THE      FAST CAT
```

DP on the “consistency matrix”

```
SeqA GARFIELD THE LAST FA-T CAT
SeqB GARFIELD THE ---- FAST CAT
```

Extended library: new pairwise alignment (AB), (AC), (AD), (BC), (BD) and (CD)

T-Coffee uses progressive strategy to derive multiple alignment

- Guide tree
- First align the closest two sequences (DP using the weights derived from the extended library)
- Align two “alignments” (using the weights from the extended library -- average over each column)
- No additional parameters (gaps etc)
    - The substitution values (weights) are derived from extended library which already considered gaps
    - High scoring segments (consistent segments) enhanced by the data set to the point that they are insensitive to the gap penalties

MUSCLE: a tool for fast MSA

- Initial progressive alignment followed by horizontal refinement (stochastic search for a maximum objective score
  - Step 1: draft progressive (using k-mer counting for fast computation of pairwise distance; tree building using UPGMA or NJ)
  - Step 2: Improved progressive to improve the tree and builds a new progressive alignment according to this tree (can be iterated).
  - Step 3: Refinement using tree-dependent restricted partitioning (each edge is deleted from the tree to divide the sequences into two disjoint subsets, from each a profile is built; the profile-profile alignment is computed, and if the score improves, retain the new alignment).
- Ref: MUSCLE: a multiple sequence alignment method with reduced time and space complexity; BMC Bioinformatics 2004, 5:113

Resources

[1] Bioinformatics Algorithms: An Active Learning Approach, P. Compeau, and P. Pevzner. Active Learning Publishers, 2nd Ed. Vol. 2, (2015) – Chapter 5

[2] http://tcoffee.vital-it.ch/cgi-bin/Tcoffee/tcoffee_cgi/index.cgi

[3] Recent evolutions of multiple sequence alignment algorithms. 2007, 3(8):e123

[4] Issues in bioinformatics benchmarking: the case study of multiple sequence alignment. Nucleic Acids Res. 2010 Jul 1