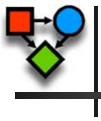


Multiple Sequence Alignment II

Lectures 20 – Dec 5, 2011
CSE 527 Computational Biology, Fall 2011

Instructor: Su-In Lee
TA: Christopher Miles

Monday & Wednesday 12:00-1:20
Johnson Hall (JHN) 022

1

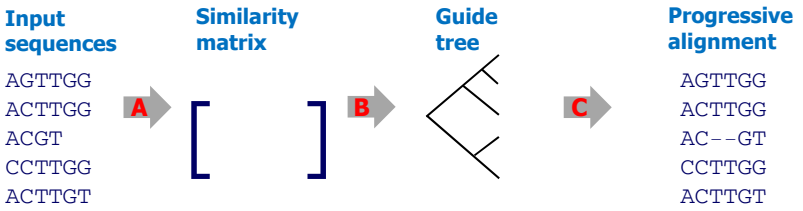
Outline

- Multiple sequence alignment methods
  - Progressive: PileUp, Clustal W (Thompson et al. 1994)
  - Iterative: MUSCLE (Edgar 2004)
  - Consistency-based: ProbCons (Do et al. 2005)

2

Review: Progressive Alignment

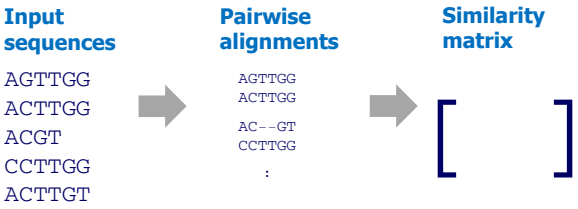
- Three basic steps shared by all progressive alignment algorithms:
  - A. Calculate a matrix of **pairwise distances** based on pairwise alignments between the sequences
  - B. Use the result of A to build a **guide tree**, which is an inferred phylogeny for the sequences
  - C. Use the tree from B to guide the **progressive alignment** of the sequences



3

(A) Calculating the Pairwise Distances

- A pair of sequences is aligned by the usual dynamic programming algorithm, and then a similarity or distance measure for the pair is calculated using the aligned portion (gaps excluded) - for example, percent identity.



4

Globin Example

DISTANCES between protein sequences:

Calculated over: 1 to 167
Correction method: Simple distance (no corrections)
Distances are: observed number of substitutions per 100 amino acids
Symmatrix version 1
Number of matrices: 1

//
Matrix 1, dimension: 7

Key for column and row indices:

1 hba_human MVLSPADKTNVKAAWGKVGAHAGEYGAEALERMFLSFPTTKTYFPHFDLSHGSAQVKGHGKKVADALTNAVAHVDDMPNALSALSDLHAHKLRVDPVNFKLLSHCLLVTLAAHLPAEFTPAVHASLDKFLASVSTVLTSK
2 hba_horse MVLSAADKTNVKAAWSKVGGHAGEYGAEALERMFLGFPTTKTYFPHFDLSHGSAQVKAHGKKVGDALTLAVGHLDDLPGALSNLSDLHAHKLRVDPVNFKLLSHCLLSTLAVHLPNDFTPAVHASLDKFLSSVSTVLTSK
3 hbb_human MVHLTPEEKSAVTALWGKVNVDEVGGEALGRLLVVYPWTQRFFESFGDLSTPDAVMGNPKVKAHGKKVLGAFSDGLAHLDNLKGTFATLSELHCDKLHVDPENFRLLGNVLVCVLAHHFGKEFTPPVQAAYQKVVAGVANALAHKYH
4 hbb_horse VQLSGEEKAAVLALWDKVNEEEVGGEALGRLLVVYPWTQRFFDSFGDLSNPGAVMGNPKVKAHGKKVLHSFGEGVHHLDNLKGTFAALSELHCDKLHVDPENFRLLGNVLVVVLARHFGKDFTPELQASYQKVVAGVANALAHKYH
5 glb5_petma MPIVDTGSVAPLSAAEKTKIRSAWAPVYSTYETSGVDILVKFFTSTPAAQEFFPKFKGLTTADQLKKSADVRWHAERIINAVNDAVASMDDTEKMSMKLRDLSGKHAKSFQVDPQYFKVLAAVIADTVAAGDAGFEKLMSMICILLRSAY
6 myg_phyca MVLSEGEWQLVLHVWAKVEADVAGHGQDILIRLFKSHPETLEKFDRFKHLKTEAEMKASEDLKKHGVTVLTALGAILKKKGHHEAELKPLAQSHATKHKIPIKYLEFISEAIIHVLHSRHPGDFGADAQGAMNKALELFRKDIAAKYKELGYQG
7 lgb2_luplu MGALTESQAALVKSSWEEFNANIPKHTHRFFILVLEIAPAAKDLFSFLKGTSEVPQNNPELQAHAGKVFKLVYEAAIQLQVTGVVVTDATLKNLGSVHVSKGVADAHFPVVKEAILKTIKEVVGAKWSEELNSAWTIAYDELAIVIKKEMNDAA

Matrix 1: Part 1

	1	2	3	4	5	6	7
1	0.00	12.06	54.68	55.40	64.12	71.74	83.57
2		0.00	55.40	53.96	64.89	72.46	82.86
3			0.00	16.44	74.26	73.94	82.52
4				0.00	75.74	73.94	81.12
5					0.00	75.91	82.61
6						0.00	80.95
7							0.00

5

(B) Building the Guide Tree

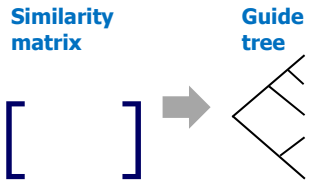
- Earlier versions of **CLUSTAL** used the unweighted pair group method using arithmetic averages (UPGMA), and this is still used in some programs.

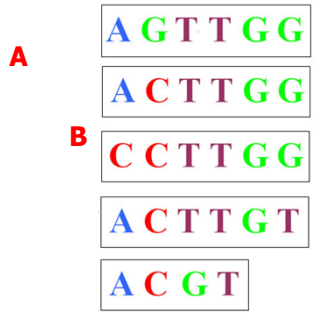
6

Unweighted Pair Group Method with Arithmetic Mean (UPGMA)

- At each step, the nearest two sequence clusters are combined into a higher-level cluster.
  - A "sequence cluster" can contain 1~N sequences.
  - No need to re-align sequences.

$$d(A,B) = \frac{1}{|A||B|}\sum_{x\in A}\sum_{y\in B} d(x,y)$$





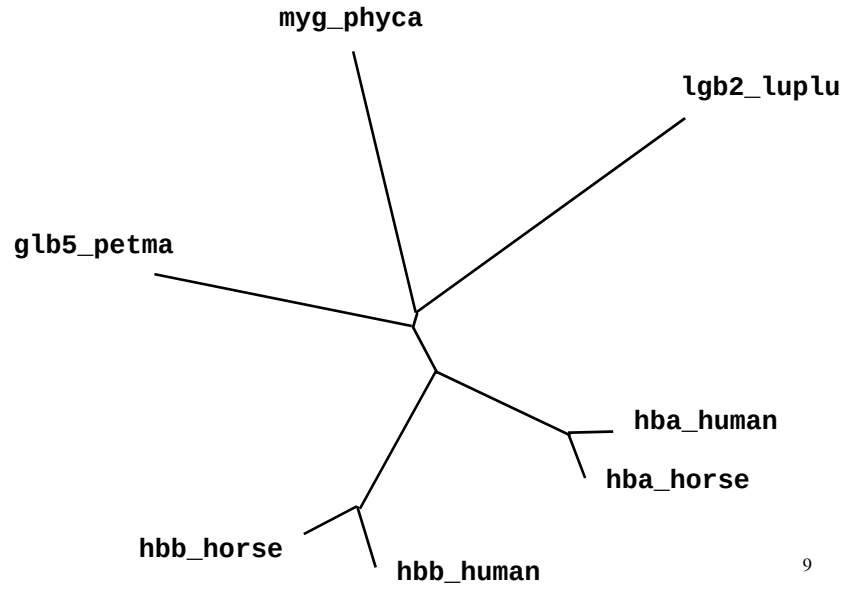
7

(B) Building the Guide Tree

- Earlier versions of **CLUSTAL** used the unweighted pair group method using arithmetic averages (UPGMA), and this is still used in some programs.
- There are many ways of building a tree from a matrix of pairwise distances. **CLUSTAL W** uses the neighbour-joining (NJ) method, which is the most favoured approach these days.
  - The **W** in **CLUSTAL W** stands for Weighted, an important feature of this program. Different weights are given to sequences and parameters in different parts of the alignment.

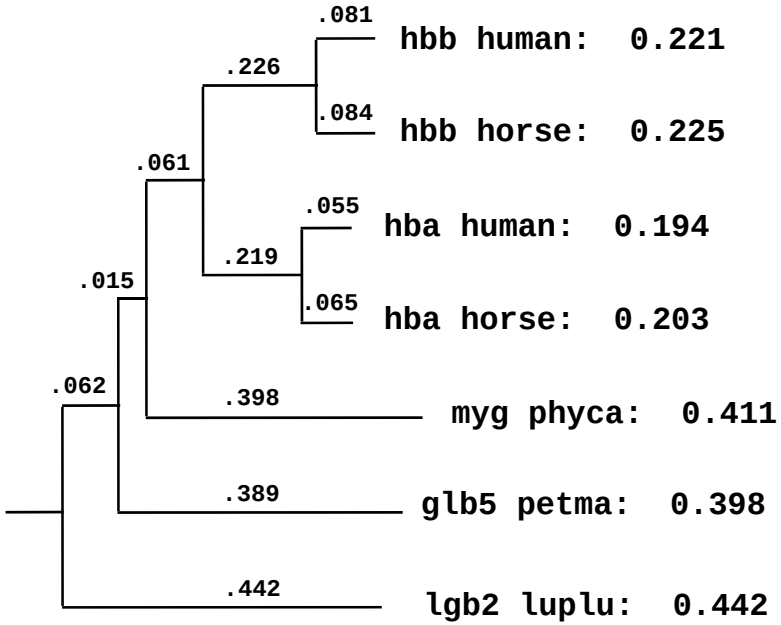
8

NJ Globin Tree



9

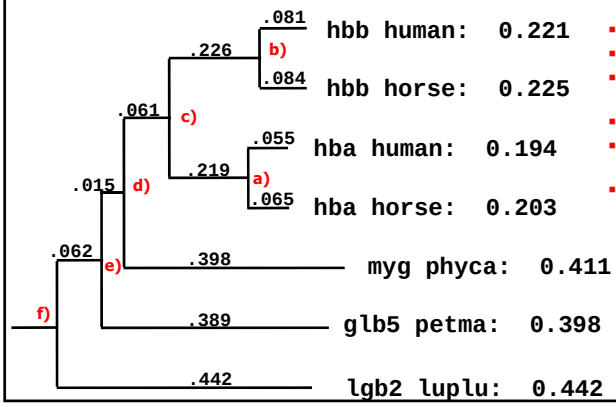
Tree, Distances and Weights Thompson et al. (1994)



10

(C) Progressive Alignment

- The basic idea is to use a series of pairwise alignments to align larger and larger groups of sequences, following the branching order of the guide tree. We proceed from the tips of the rooted tree towards the root.
- In our globin example, we align in the following order:



- a) human and horse alpha-globin;
- b) human and horse beta-globin;
- c) the two alppha-globins and the two beta-globins;
- d) myoglobin and the hemoglobins;
- e) cyanohemoglobin and the combined haemoglobin, myoglobin group;
- f) leghemoglobin and the rest.

11

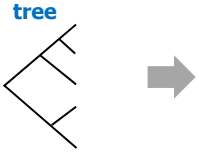
(C) Progressive Alignment

- At each stage a full dynamic programming algorithm is used, with a residue scoring matrix (e.g., a PAM or a BLOSUM matrix) and gap opening and extension penalties.
- Each step consists of aligning two existing alignments. Scores at a position are averages of all pairwise scores for residues in the two sets of sequences.

Input sequences

```
AGTTGG
ACTTGG
ACGT
CCTTGG
ACTTGT
```

Guide tree



Progressive alignment

```
AGTTGG
ACTTGG
AC--GT
CCTTGG
ACTTGT
```

12

Scoring an Alignment of Two Partial Alignments

- What is score s(tlkk,vi)?

$$\frac{1}{8}[M(t,v)w_1w_5 + M(t,i)w_1w_6 + \ldots + M(k,i)w_4w_6]$$

```
1      peeksavtal
2      geekaavlal
3      padktnvkaa
4      aadktnvkaa

5      egewqlvlhv
6      aaektkirsa
```

Sequence weights
$w_1, \ldots, w_6$

Progressive Alignment

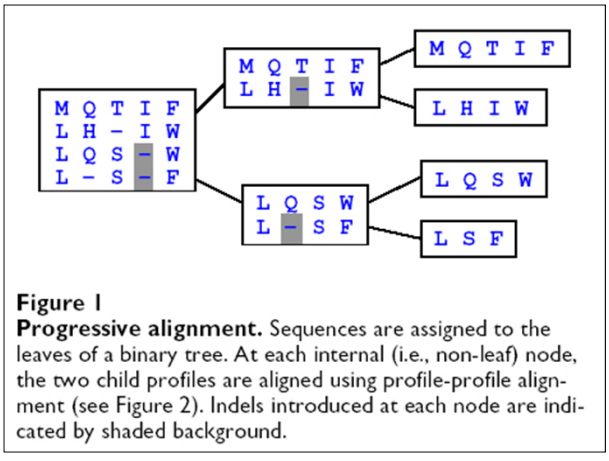


Figure 1
Progressive alignment. Sequences are assigned to the leaves of a binary tree. At each internal (i.e., non-leaf) node, the two child profiles are aligned using profile-profile alignment (see Figure 2). Indels introduced at each node are indicated by shaded background.

Profile-Profile Alignment

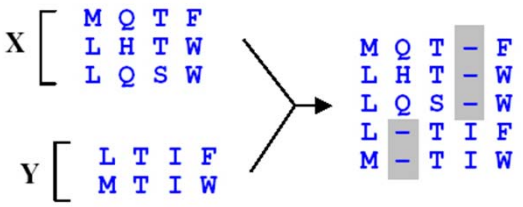


Figure 2
Profile-profile alignment. Two profiles (multiple sequence alignments) X and Y are aligned to each other such that columns from X and Y are preserved in the result. Columns of indels (gray background) are inserted as needed in order to align the columns to each other. The score for aligning a pair of columns is determined by the profile function, which should assign a high score to pairs of columns containing similar amino acids.

15

(C) Progressive alignment

- At each stage a full dynamic programming algorithm is used, with a residue scoring matrix (e.g., a PAM or a BLOSUM matrix) and gap opening and extension penalties.
- Each step consists of aligning two existing alignments. Scores at a position are averages of all pairwise scores for residues in the two sets of sequences.
- Gaps that are present in older alignments remain fixed. New gaps introduced at each stage initially get full opening and extension penalties, even if inside old gap positions.

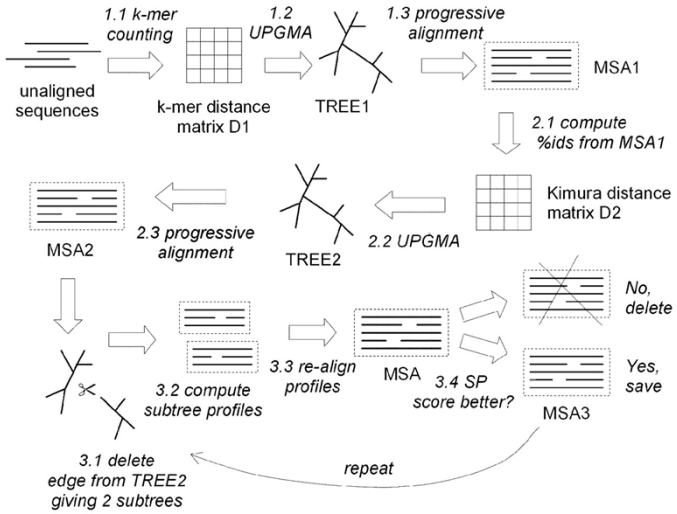
16

Outline

- Multiple sequence alignment methods
  - Progressive: PileUp, Clustal W (Thompson et al. 1994)
  - Iterative: MUSCLE (Edgar 2004)
  - Consistency-based: t-Coffee (Notredame 2000), ProbCons (Do et al. 2005)

17

MUSCLE Algorithm: Using The Iteration



MUSCLE – Overview

- Basic Idea: A progressive alignment is built, to which horizontal refinement is applied
- 3 stages of the algorithm
  - At the completion of each, a multiple alignment is available and the algorithm can be terminated

19

The Algorithm (1)

- Stage 1: Draft Progressive – *Builds a progressive alignment*
  - Similarity of each pair of sequences is computed using
    - *k*-mer counting – similar sequences have similar *k*-mer counts
    - The *k*-mer distance is computed for each pair of input sequences, giving distance matrix D1
    - D1 is clustered by UPGMA, producing binary tree TREE1
  - A tree is constructed and a root is identified
  - A progressive alignment is built by following the branching order of TREE1, yielding a multiple alignment
  - The main source in the draft progressive stage is the approximate *k*-mer distance measure
    - Needs refinement

20

The Algorithm (2)

- Stage 2: Improved Progressive – *Improves the tree*
  - Similarity of each pair of sequences is computed using fractional identity from the mutual alignment from Stage 1, constructing distance matrix D2
  - TREE2 is constructed by applying a clustering method to D2
  - The trees are compared; a set of nodes for which the branching order has changed is identified
  - A new alignment is built, the existing one is retained if the order is unchanged

21

Tree Comparison

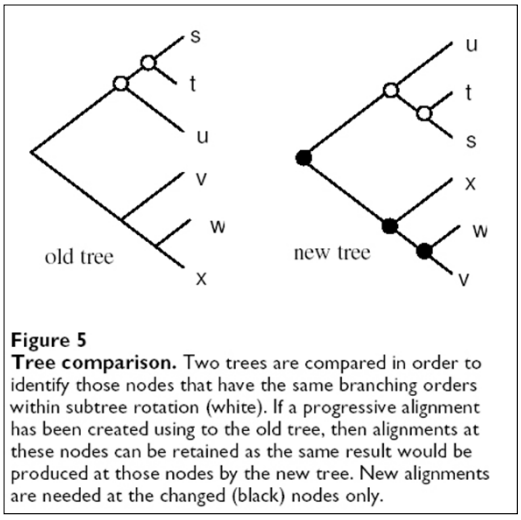


Figure 5
Tree comparison. Two trees are compared in order to identify those nodes that have the same branching orders within subtree rotation (white). If a progressive alignment has been created using to the old tree, then alignments at these nodes can be retained as the same result would be produced at those nodes by the new tree. New alignments are needed at the changed (black) nodes only.

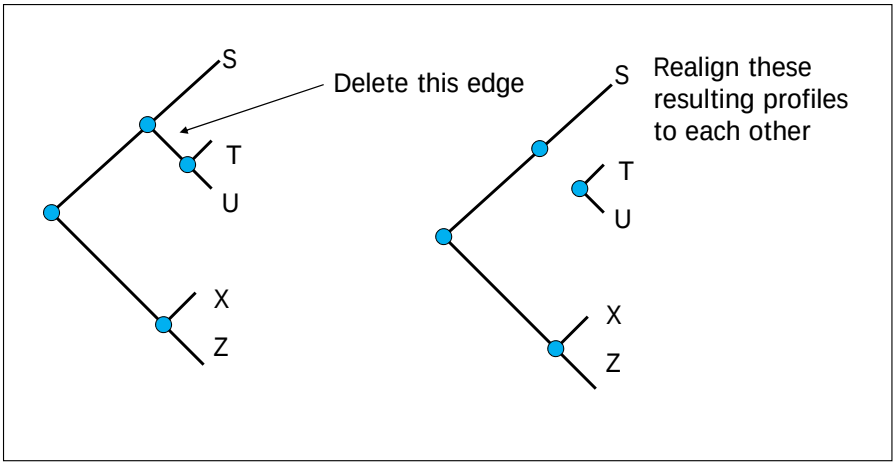
22

The Algorithm

- Stage 3: Refinement – *Iterative Refinement is performed*
  - An edge is deleted from TREE2, dividing the sequences into two disjoint subsets
  - The profile (MA) of each subset is extracted
  - The profiles are re-aligned to each other
  - The score is computed, if the score has increased, the alignment is retained, otherwise it is discarded
  - Algorithm terminates at convergence

23

Iterative Refinement



24

MUSCLE Algorithm

- Improvements in selection of heuristics
- Close attention paid to implementation details
- Higher accuracy than progressive alignment methods
- http://www.drive5.com/muscle

- References
  - Edgar, Robert C. (2004), MUSCLE: multiple sequence alignment with high accuracy and high throughput, *Nucleic Acids Research* **32**(5), 1792-97
  - Edgar, Robert C (2004), MUSCLE: a multiple sequence alignment method with reduced time and space complexity. *BMC Bioinformatics*, **5**(1):113

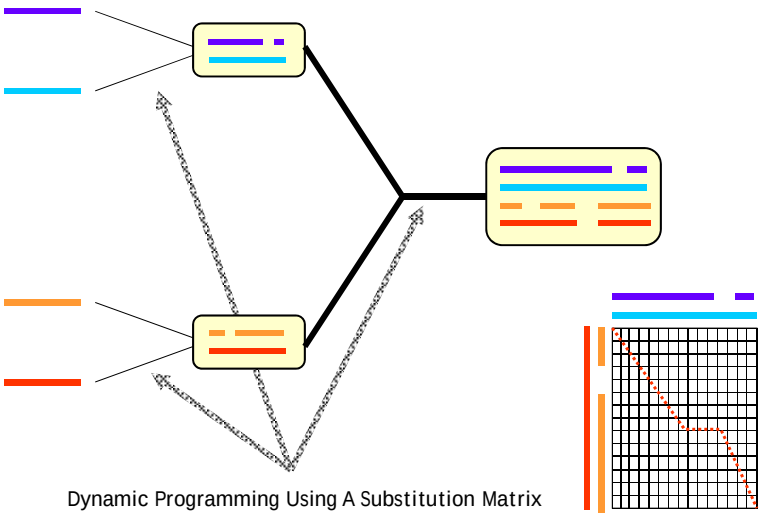
25

Outline

- Multiple sequence alignment methods
  - Progressive: PileUp, Clustal W (Thompson et al. 1994)
  - Iterative: MUSCLE (Edgar 2004)
  - Consistency-based: ProbCons (Do et al. 2005)

26

Progressive Alignment

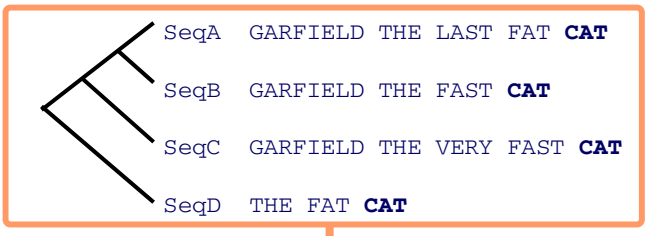


Dynamic Programming Using A Substitution Matrix

27

Progressive Alignment Principle and Its Limitation

- Regular progressive alignment strategy:



- Final alignment

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

28

The Extended Alignment Principle

```
SeqA  GARFIELD THE LAST FAT CAT      Prim. Weight = 88
SeqB  GARFIELD THE FAST CAT ---

SeqA  GARFIELD THE LAST FA-T CAT     Prim. Weight = 77
SeqC  GARFIELD THE VERY FAST CAT

SeqA  GARFIELD THE LAST FAT CAT      Prim. Weight = 100
SeqD  -------- THE ---- FAT CAT

SeqB  GARFIELD THE ---- FAST CAT     Prim. Weight = 100
SeqC  GARFIELD THE VERY FAST CAT

SeqB  GARFIELD THE FAST CAT          Prim. Weight = 100
SeqD  -------- THE FA-T CAT

SeqC  GARFIELD THE VERY FAST CAT     Prim. Weight = 100
SeqD  -------- THE ---- FA-T CAT
```

29

The Extended Alignment Principle

```
SeqA  GARFIELD THE LAST FAT CAT      Prim. Weight = 88
      ||||||||  |||  ||||  |||
SeqB  GARFIELD THE FAST CAT ---

SeqA  GARFIELD THE LAST FA-T CAT     Prim. Weight = 77
      ||||||||  |||  ||||  ||  |  |||
SeqC  GARFIELD THE VERY FAST CAT
      ||||||||  |||       ||||  |||
SeqB  GARFIELD THE ---- FAST CAT

SeqA  GARFIELD THE LAST FA-T CAT     Prim. Weight = 100
                |||       ||  |  |||
SeqD            THE       FA-T CAT
                |||       ||  |  |||
SeqB  GARFIELD THE ---- FAST CAT
```


SeqA GARFIELD THE LAST FAT CAT
SeqB GARFIELD THE FAST CAT ---

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

SeqA GARFIELD THE LAST FAT CAT
SeqD -------- THE ---- FAT CAT

SeqB GARFIELD THE ---- FAST CAT
SeqC GARFIELD THE VERY FAST CAT

SeqB GARFIELD THE FAST CAT
SeqD -------- THE FA-T CAT

SeqC GARFIELD THE VERY FAST CAT
SeqD -------- THE ---- FA-T CAT

The Extended Alignment Principle

- Pairwise alignment (dynamic programming)

```
SeqA  GARFIELD THE LAST FAT CAT      Prim. Weight = 88
      ||||||||| ||| |||| |||
SeqB  GARFIELD THE FAST CAT ---
```

- Extended library

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

Dynamic Programming

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

Can we take into account the "consistency" when evaluating a pairwise alignment?

31

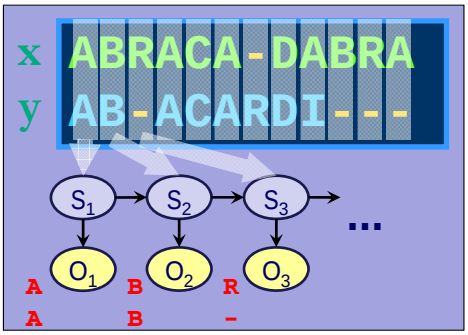
ProbCons – Overview

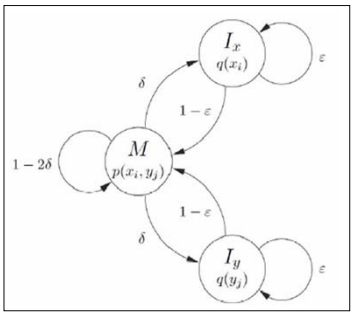
- Alignment generation can be directly modeled as a first order Markov process involving state emission and transitions
- ProbCons is a pair-hidden Markov model-based progressive alignment algorithm that primarily differs from most typical approaches in its use of:
  - Maximum expected accuracy
  - Probabilistic consistency transformation as a scoring function
    - To incorporate multiple sequence conservation information during pairwise alignment
- Model parameters obtained using unsupervised maximum likelihood methods

32

ProbCons – Hidden Markov Model

- ProbCons uses the HMM to specify the probability distribution over all alignments between a pair of sequences.

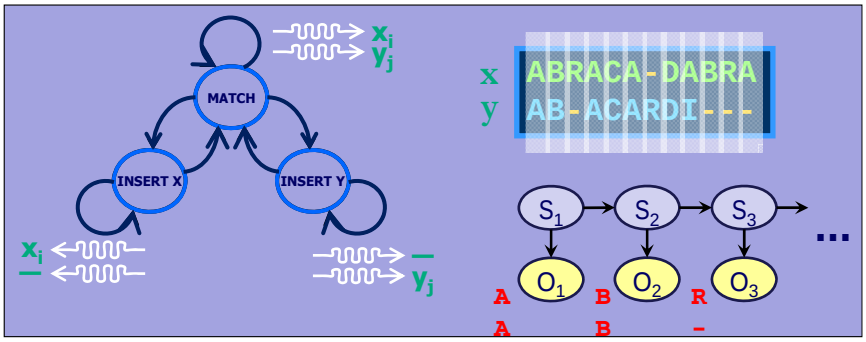




- Deletion penalties on Match → Gap transitions
- Extension penalties on Gap → Gap transitions
- Match/Mismatch penalties on Match emissions

33

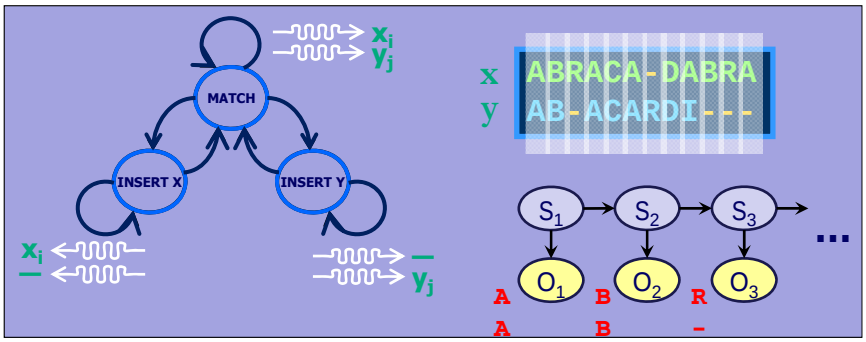
ProbCons – Hidden Markov Model



- Basic HMM for sequence alignment between two sequences
  - M emits two letters, one from each sequence
  - I_x emits a letter from x that aligns to a gap
  - I_y emits a letter from y that aligns to a gap

34

ProbCons – Hidden Markov Model

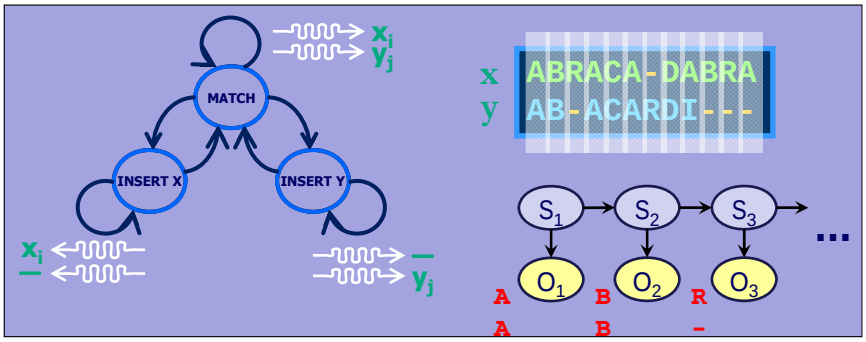


- For an alignment ***a***,

$$\mathbf{P}(a|x,y) = \pi(s_1)\left(\prod_{i=1}^{n-1}\alpha(s_i \to s_{i+1})\right)\left(\prod_{i=1}^{n}\beta(o_i|s_i)\right)$$

35

ProbCons – Hidden Markov Model



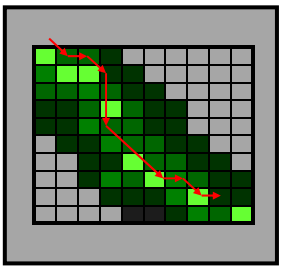
- Let ***a**** be the (unknown) alignment that most nearly represents the "true" biological alignment,
  - Formally, the posterior probability

$$\mathbf{P}(x_i \sim y_j \in a^*|x,y) = \sum_{a \in A}\mathbf{P}(a|x,y)\mathbf{1}\{x_i \sim y_j \in a\}$$

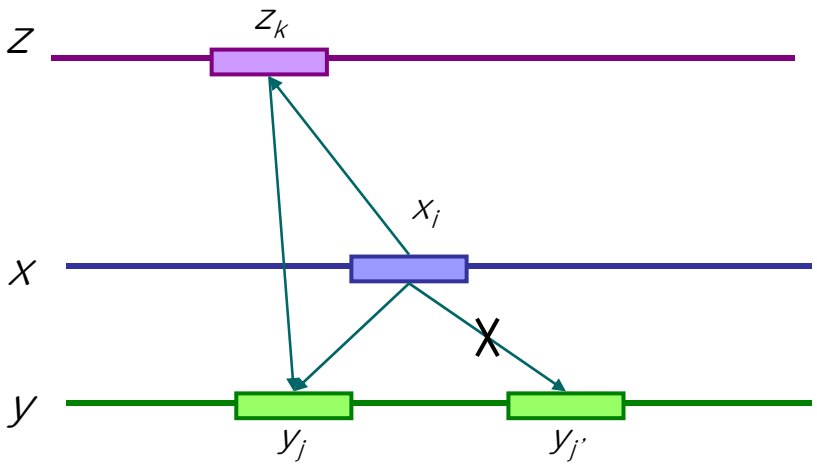
36

ProbCons – Computing Maximum Expected Accuracy (MEA)

- Define accuracy (a, a*) = the expected number of correctly aligned pairs of letters divided by the length of the shorter sequence
- The MEA alignment is found by finding the highest summing path through the matrix
- $M_{xy}[i, j] = P(x_i \text{ is aligned to } y_j | x, y)$
  - We can efficiently compute these
  - just need to compute these terms!
  - Can use dynamic programming



Probabilistic Consistency



38

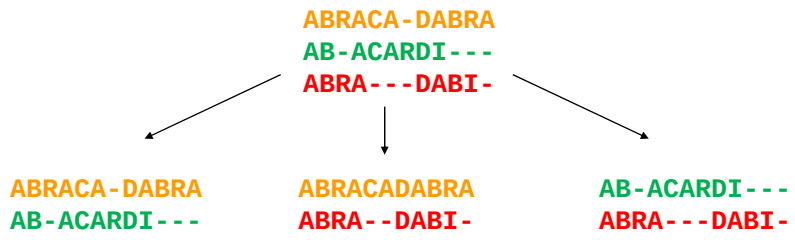
Probabilistic Consistency

- Compute P(~~x_i is aligned to y_j | x, y~~)
  P(x_i is aligned to y_j | x, y, z)
- We can re-estimate M_{xy} as $(M_{xz})\cdot(M_{zy})$ where z is a third sequence to which x and y are aligned
- $M_{xy}[i,j] = \sum_{k=1}^{n} M_{xz}[i,k] \cdot M_{zy}[k,j]$, where n is length of z
  - Complexity $O(L^3)$, but we can reduce computation by transforming the M_{xz} and M_{zy} into sparse matrices by discarding all values smaller than a threshold w.
- We follow the alignment from position i of x, to position j of y, through all intermediate positions k of a third sequence z

39

ProbCons – Multiple Alignment

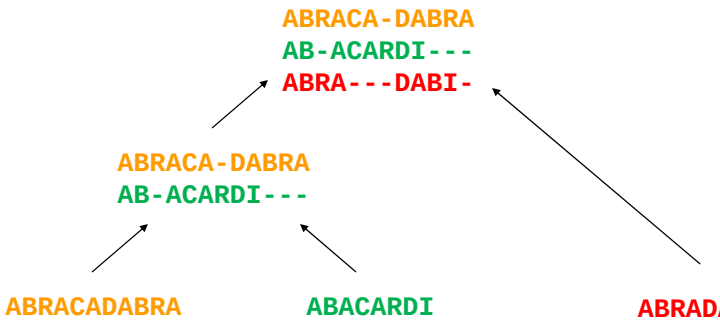
- A straightforward generalization
  - Sum-of-pairs



40

ProbCons – Multiple Alignment

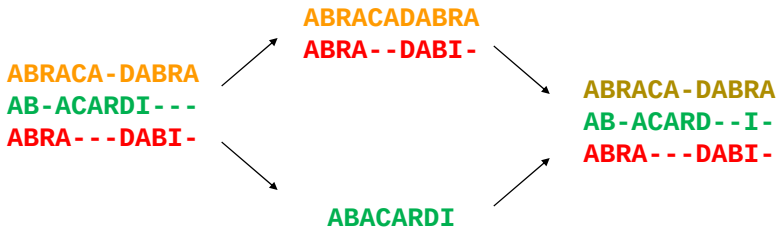
- A straightforward generalization
  - Sum-of-pairs
  - Tree-based progressive alignment



41

ProbCons – Multiple Alignment

- A straightforward generalization
  - Sum-of-pairs
  - Three-based progressive alignment
  - Iterative refinement



42

ProbCons – The Algorithm I

- Step 1: Computation of posterior-probability matrices
  - For every pair of sequences x and y, compute the probability that letters x_i, y_j are paired in **a***, an alignment of x and y that is randomly generated by the model

$$\mathbf{P}(x_i \sim y_j \in a^* | x,y) = \sum_{a \in A} \mathbf{P}(a|x,y)\mathbf{1}\{x_i \sim y_j \in a\}$$

- Step 2: Computation of expected accuracies
  - Define the expected accuracy of a pairwise alignment $\mathbf{a_{xy}}$ to be the expected number of correctly aligned pairs of letters divided by the length of the shorter sequence
  - Compute the alignment $\mathbf{a_{xy}}$ that maximizes expected accuracy E(x,y) using dynamic programming

43

ProbCons – The Algorithm II

- Step 3: Probabilistic consistency transformation
  - Re-estimate the scores with probabilistic consistency transformation by incorporating similarity of x and y to other sequences into the pairwise comparison of x and y
  - Computed efficiently using sparse matrix multiplication ignoring all entries smaller than some threshold
- Step 4: Computation of a guide tree
  - Construct a tree by hierarchical clustering using E(x,y)
  - Cluster similarity is defined by a weighted average of pairwise similarities between the clusters

44

ProbCons – The Algorithm III

- Step 5: Progressive alignment
  - Align sequence groups hierarchically according to the order specified in the guide tree
  - Score using a sum of pairs function in which the aligned residues are scored according to the match quality scores and the gap penalties are set to 0
- Step 6: Iterative refinement
  - Randomly partition alignment into two groups of sequences and realign
  - May be repeated as necessary

45

ProbCons

- Results
  - Best results so far
  - Longer in running time due to the computation of posterior probability matrices (Step 1)
  - Doesn't incorporate biological information
  - Could provide improved accuracy in DNA multiple alignment
  - http://probcons.stanford.edu

46