15-499: Algorithms and Applications

Computational Biology III
- Multiple Sequence Alignment

Multiple Alignment

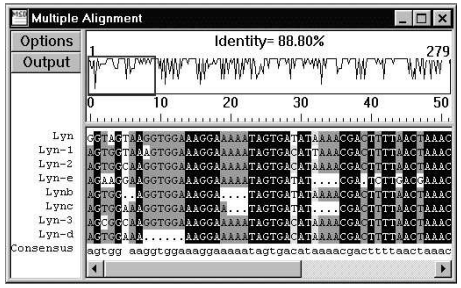
```
A C T _ G T A
A C A C G T T
A G T G _ T A
C C _ G C T A
```

Goal: match the "maximum" number of aligned pairs of symbols.

Applications:
- Assembling multiple noisy reads of fragments of sequences
- Finding a canonical among members of a family and studying how the members differ

The problem is NP-hard

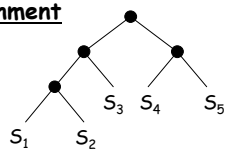
Example Output



Output from typical multiple alignment software DNAMAN (using ClustalW)

Scoring Multiple Alignments

1. **Distance from consensus S_c:** $D = \sum_{S_i \in S} D(S_i, S_c)$
2. **Pairwise distances:** $D = \sum_{S_i \in S} \sum_{S_j \in S / S_i} D(S_i, S_j)$
3. **Evolutionary Tree Alignment**



$$D = D(S_1, S_2) + D(S_4, S_5) + D(S_{12}, S_3) + D(S_{123}, S_{45})$$

Approaches

Dynamic programming: optimal, but takes time that is exponential in *p*

Center Star Method: approximation

Clustering Methods: also called iterative pairwise alignment. Typically an approximation. Many variants, many software packages

Using Dynamic Programming

For *p* sequences of length *n* we can fill in a *p*-dimensional array in n^p time and space.

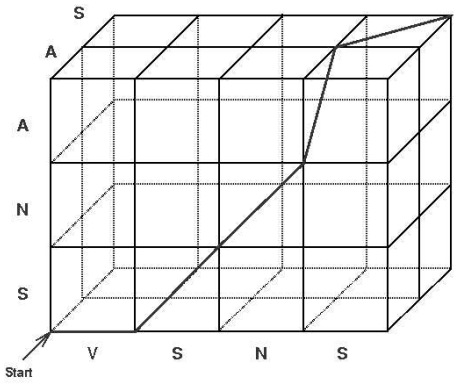
For example for $p = 3$:

$$D_{ijk} = \min \begin{cases} D_{i-1,j-1,k-1} + d(a_i, b_j, c_k) \\ D_{i-1,j-1,k} + d(a_i, b_j, \_) \\ D_{i-1,j,k} + d(a_i, \_, \_) \\ \ldots \end{cases}$$ 7 cases

where $d(a,b,c) = d(a,b) + d(b,c) + d(a,c)$ assuming the pairwise distance metric.

Takes time exponential in *p*. Perhaps OK for $p = 3$

Example



V S N _ S
_ S N A _
_ _ _ A S

Optimization

As in the case of pairwise alignment we can view the array as a graph and find shortest paths.

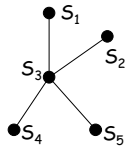
Used in a program called MSA.

Can align 6 strings consisting of 200 bp each in a "practical" amount of time.

Center Star Method

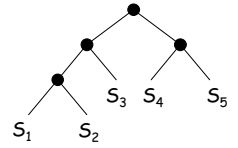
1. Find $S_t \in S$ minimizing $\sum_{S_i \in S/S_t} D(S_i, S_t)$
2. Add remaining sequences S/S_t one by one so alignment of each is optimal wrt S_t.
   Add spaces if needed

Time: $O(p^2n^2)$

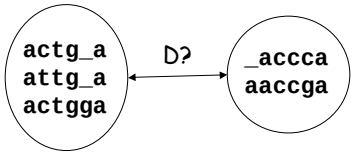


Using Clustering

1. Compute $D(S_i,S_j)$ for all pairs
2. Bottom up cluster
   I. All sequences start as their own cluster
   II. Repeat
      a) find the two "closest" clusters and join them into one
      b) Find best alignment of the two clusters being joined



Distances between Clusters



Could use difference between consensus.

A popular technique is called the "Unweighted Pair-Group Method using arithmetic Averages" (UPGMA).
It takes the average of all distances among the two clusters.

Implemented in Clustal and Pileup

Summary of Matching

Types of matching:
- **Global**: align two sequences A and B
- **Local**: align A with any part of B
- **Multiple**: align k sequences (NP-complete)

Cost models
- **LCS and MED**
- **Scoring matrices:** Blosum, PAM
- **Gap cost:** affine, general

Methods
- **Dynamic programming:** many optimizations
- **"Fingerprinting"**: hashing of small seqs. (approx.)
- **Clustering:** for multiple alignment (approx.)