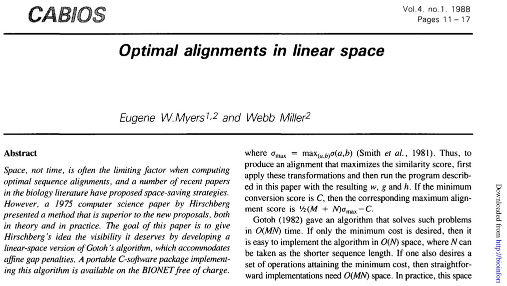
Algorithmen der Bioinformatik I
WS 2017/2018

Burkhard Morgenstern
Peter Meinicke

Dept. Bioinformatics
Institute of Microbiology and Genetics (IMG)
University of Göttingen

November 6, 2017

Alignment with linear memory


CABIOS Vol.4. no.1. 1988 Pages 11 – 17

Optimal alignments in linear space

Eugene W.Myers[1,2] and Webb Miller[2]

Abstract

Space, not time, is often the limiting factor when computing optimal sequence alignments, and a number of recent papers in the biology literature have proposed space-saving strategies. However, a 1975 computer science paper by Hirschberg presented a method that is superior to the new proposals, both in theory and in practice. The goal of this paper is to give Hirschberg's idea the visibility it deserves by developing a linear-space version of Gotoh's algorithm, which accommodates affine gap penalties. A portable C-software package implementing this algorithm is available on the BIONET free of charge.

where $\sigma_{\max} = \max_{(a,b)}\sigma(a,b)$ (Smith *et al.*, 1981). Thus, to produce an alignment that maximizes the similarity score, first apply these transformations and then run the program described in this paper with the resulting w, g and h. If the minimum conversion score is C, then the corresponding maximum alignment score is $\frac{1}{2}(M + N)\sigma_{\max} - C$.

Gotoh (1982) gave an algorithm that solves such problems in $O(MN)$ time. If only the minimum cost is desired, then it is easy to implement the algorithm in $O(N)$ space, where N can be taken as the shorter sequence length. If one also desires a set of operations attaining the minimum cost, then straightforward implementations need $O(MN)$ space. In practice, this space

Figure: Application of Hirschberg's linear-space algorithm to pairwise alignment: G. Myers and W. Miller, 1988, *Optimal alignments in linear space*

Alignment with linear memory

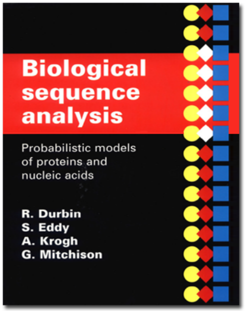

Biological sequence analysis
Probabilistic models of proteins and nucleic acids
R. Durbin
S. Eddy
A. Krogh
G. Mitchison

Figure: Alternative algorithm: R. Durbin *et al.*, 1998, *Biological Sequence Analysis*, pp. 34-35

Alignment with linear memory

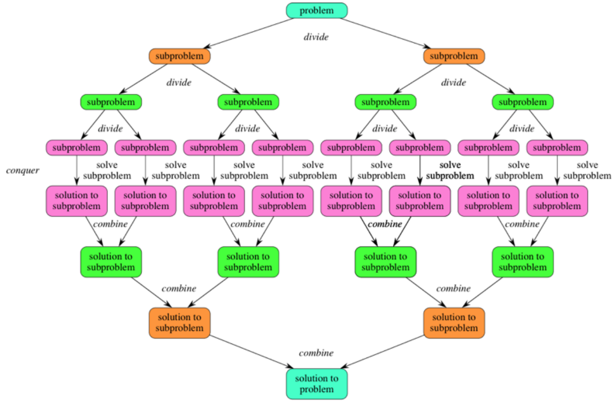


Figure: Divide-and-conquer principle (`https://www.khanacademy.org/`).

Alignment with linear memory

Idea (Divide and Conquer for optimal alignment):

- Find position in *middle* column of DP matrix through which optimal path (alignment) runs (‘optimal midpoint’).
- Only upper-left and lower-right sections of DP matrix have to be considered to find optimal path.
- Calculte optimal path for upper-left and lower-right sub-matrices independently.
- Repeat recursively until search space small enough.

Alignment with linear memory

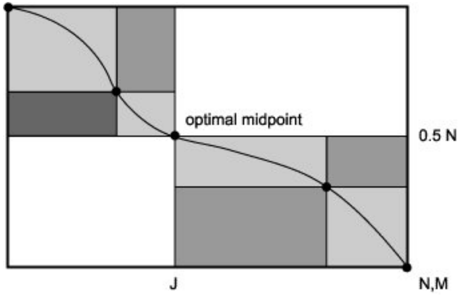


Figure: Finding optimal midpoint in DP matrix. (D. Mount, *Bioinformatics*)

Alignment with linear memory

In more detail: for sequences of length m and n, consider

$$m' = \left\lfloor \frac{m}{2} \right\rfloor$$

To find optimal ‘midpoint’ in middle column, calculate for (i, j) with $i \geq m'$

$$c(i, j)$$

such that optimal path from $(0, 0)$ to (i, j) goes through

$$(m', c(i, j))$$

Idea: use *DP* algorithm; calculate $c(i, j)$ together with $d(i, j)$

Alignment with linear memory

$c(i, j)$ depends on where maximum is obtained in recursion formula

$$F(i,j) = \max \begin{cases} F(i-1, j-1) & + & s(X_i, Y_j) \\ F(i-1, j) & - & g \\ F(i, j-1) & - & g \end{cases}$$

Depending on where maximum is obtained:

$$c(i,j) = \begin{cases} c(i-1, j-1) & \text{if maximum in 1st line} \\ c(i-1, j) & \text{if maximum in 2nd line} \\ c(i, j-1) & \text{if maximum in 3rd line} \end{cases}$$

$\Rightarrow$ Optimal path from $(0, 0)$ to (m, n) goes through $(m', c(m, n))$.

Alignment with linear memory

Complexity of algorithm:

- *Memory* complexity linear
- *Time* complexity:
  - 1. step takes $m \cdot n$ time
  - 2. step takes $\frac{1}{2} \cdot m \cdot n$ time
  - k-th step takes $2^{-k} \cdot m \cdot n$ time

$\Rightarrow$ for k steps, algorithm takes

$$\sum_{0}^{k} 2^{-i} \cdot m \cdot n \to 2 \cdot m \cdot n$$

time

Phylogeny Reconstruction

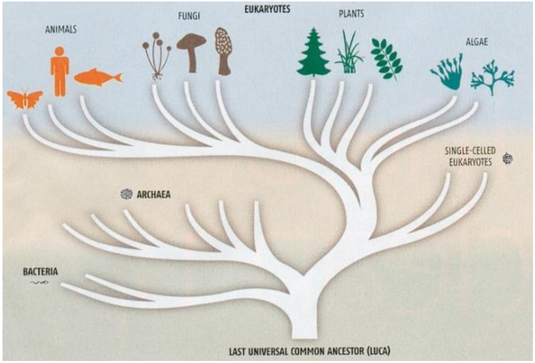


Figure: The tree of life (source: `http://universe-review.ca`)

Phylogeny Reconstruction

Goal: phylogeny reconstruction for

- Species
- Molecules (e.g. protein families)

Data basis: *sequence data* (protein or nucleic acid).

- Given: sequence family with multiple sequence alignment
- Wanted: tree – with or without ‘root’ – that ‘fits’ sequence data

Phylogeny Reconstruction

```
seq1    R Y L V M R E A Q E W
seq2    Y I M Q E V Q Q E R A
seq3    W T Y L V M E A Q Y E S A Q
seq4    A L Y I A M E V Q Y E S A
```

Phylogeny Reconstruction

```
seq1    - R Y L V M R E A Q - E W - -
seq2    - - Y I - M Q E V Q Q E R A -
seq3    W T Y L V M - E A Q Y E S A Q
seq4    A L Y I A M - E V Q Y E S A -
```

Phylogeny Reconstruction

Approaches:

(1) Distance methods:
Calculate pairwise distances between species/sequences.
Find tree that represents distances.

(2) Maximum Parsimony:
Find tree that minimizes number of mutations

(3) Probabilistic methods:
- Maximum Likelihood.
- Bayesian methods.

Distance Methods for Tree Reconstruction

Input: distance matrix with pairwise distances between OTUs (‘Operational Taxonomic Units’, = species, sequences *etc.*)

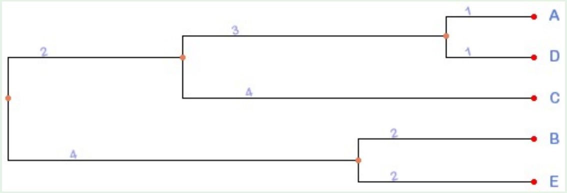
Output: tree with OTUs at the leaves that ‘represents’ input distances.

More precisely: length of paths between leaves are equal to distances in matrix.

Distance Methods for Tree Reconstruction

Example (Distance matrix and tree)

	A	*B*	*C*	*D*	*E*
A	0	12	8	2	12
B		0	12	12	4
C			0	8	12
D				0	12
E					0



Distance Methods for Tree Reconstruction

For a given distance matrix:

How do we know if there is a tree representing the distances?

Simplest case: all leaves have same distance from ‘root’.

Distance Methods for Tree Reconstruction

Definition (Ultrametric)

A distance function d on a set X is called ultrametric *if it satisfies*

$$\begin{aligned} d(x,y) &\geq 0 \\ d(x,y) &= 0 \Leftrightarrow x = y \\ d(x,y) &= d(y,x) \\ d(x,z) &\leq \max\{d(x,y), d(y,z)\} \end{aligned}$$

for all x, y, z.

Last line equivalent to: two of the distances $d(x,y), d(y,z), d(x,z)$ are equal, and the third one is smaller or equal than these two.

Distance Methods for Tree Reconstruction

Definition (Ultrametric tree)

A rooted tree is called ultra metric tree*, if all distances from the root to the leaves are equal.*

Theorem

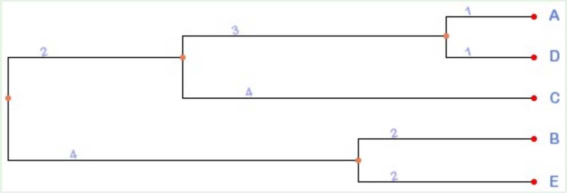
If a distance matrix M has the ultrametric *property, then there exists an ultrametric tree representing the distances from M.*

For *ultrametric* $n \times n$ distance matrix, clustering method *UPGMA* finds 'correct' tree representing distances.

Distance Methods for Tree Reconstruction

Example (Ultrametric distance matrix and tree)

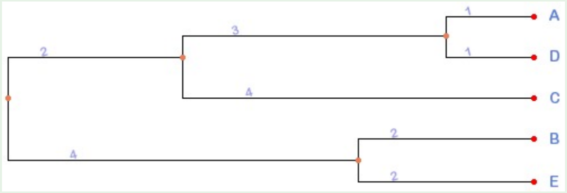
	A	*B*	*C*	*D*	*E*
A	0	12	8	2	12
B		0	12	12	4
C			0	8	12
D				0	12
E					0



Distance Methods for Tree Reconstruction

Example (Ultrametric distance matrix and tree)

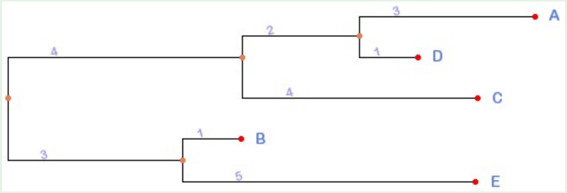
	A	*B*	*C*	*D*	*E*
A	0	12	8	2	12
B		0	12	12	4
C			0	8	12
D				0	12
E					0



Distance Methods for Tree Reconstruction

Example (*Non-ultrametric* distance matrix and tree)

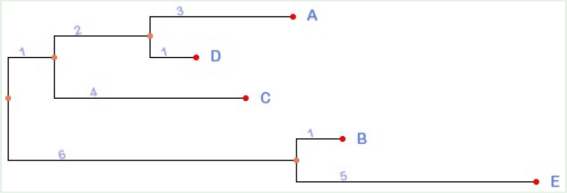
	A	*B*	*C*	*D*	*E*
A	0	13	9	4	17
B		0	12	11	6
C			0	7	16
D				0	15
E					0



Distance Methods for Tree Reconstruction

Example (Same distance matrix, tree with different root position)

	A	*B*	*C*	*D*	*E*
A	0	13	9	4	17
B		0	12	11	6
C			0	7	16
D				0	15
E					0



Distance Methods for Tree Reconstruction

Result:

For non-ultrametric distance matrix, tree can be found, but position of root not clear.

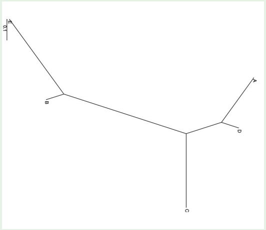
$\Rightarrow$ Find tree *without* root.

For given distance matrix, *Neighbour Joining* finds tree representing input distances – if such a tree exists.

Distance Methods for Tree Reconstruction

Example (Distance matrix and *unrooted* tree)

	A	B	C	D	E
A	0	13	9	4	17
B		0	12	11	6
C			0	7	16
D				0	15
E					0



How to find a root for the tree?

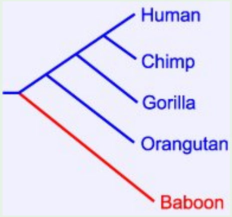
Some phylogeny methods produce *unrooted* trees.

Use *outgroup* to define root.

- Chose organism (or group of organisms), that are *not* part of the group under study, but not too far away.
- Construct tree for group under study *and* outgroup, define branching point of the two groups as root for group under study.

How to find a root for the tree?

Example (Outgroup to find root for unrooted tree)



Baboon used as 'outgroup' to study phylogeny of Human, Chimp, Gorilla, Orangutan