582670 Algorithms for Bioinformatics

Lecture 6: Distance based clustering and phylogeny

3.10.2013

Adapted from slides by Leena Salmena and Veli Mäkinen, which are partly from http://bix.ucsd.edu/bioalgorithms/slides.php.

Outline

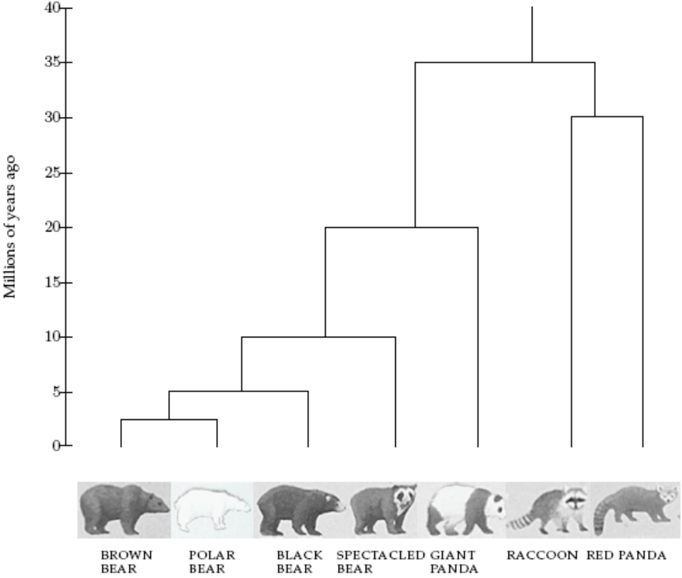
Distance-based clustering, UPGMA

Neighbor joining

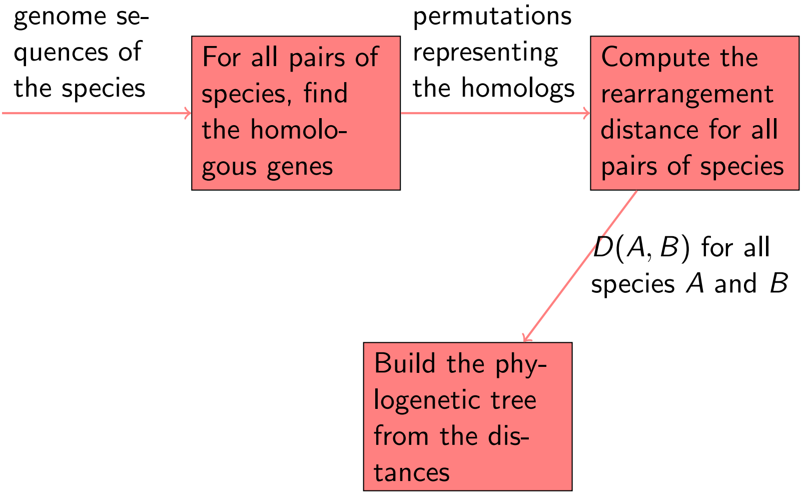
Study group assignments

About the exam

Phylogenetic tree: Bears



Phylogeny by distance method pipeline



genome sequences of the species

For all pairs of species, find the homologous genes

permutations representing the homologs

Compute the rearrangement distance for all pairs of species

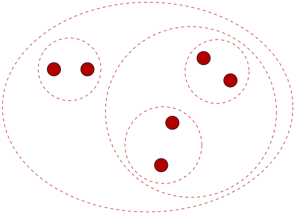
$D(A, B)$ for all species A and B

Build the phylogenetic tree from the distances

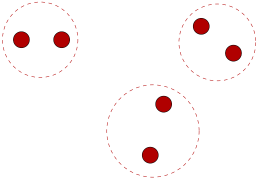
Clustering

- *Clustering* can be loosely stated as the problem of grouping objects into sets called clusters, where the members of the cluster are similar in some sense.

- *Hierarchical clustering*:
  - Iteratively join two closest clusters forming a tree hierarchy (agglomerative... also divisive version exists)
  - Distance between clusters can be e.g. max pair-wise distance (complete linkage), min (single linkage), UPGMA (average linkage), neighbor joining

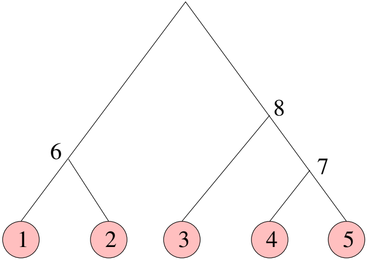


- *Partitional clustering:*
  - *k*-means



Distances in a phylogenetic tree

- Distance matrix $D = (d_{ij})$ gives pairwise distances for *leaves* of the phylogenetic tree
- In addition, the phylogenetic tree will now specify distances between internal nodes
  - Denote these with d_{ij} as well



Distance d_{ij} states how far apart species i and j are evolutionary.

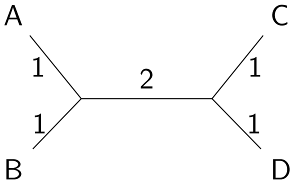
Distances in evolutionary context

- Distance d_{ij} in evolutionary context satisfy the following conditions:
  - Positivity: $d_{ij} \geq 0$
  - Identity: $d_{ij} = 0$ if and only if $i = j$
  - Symmetry: $d_{ij} = d_{ji}$ for each i, j
  - Triangle inequality: $d_{ij} \leq d_{ik} + d_{kj}$ for each i, j, k
- Distance satisfying these conditions is called *metric*
- In addition, evolutionary mechanisms may impose additional constraints on the distances
  - *additive* and *ultrametric* distances

Additive trees

- Suppose that every edge in a tree is labeled with a distance d_{ij}
- A tree is called *additive* if for every pair of leaves the distance between the leaves is the sum of the edge distances on the path between the leaves.
- Example:

	A	B	C	D
A	0	2	4	4
B	2	0	4	4
C	4	4	0	2
D	4	4	2	0

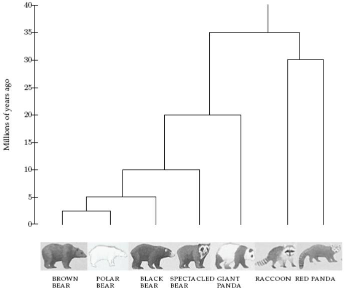


Ultrametric trees

- A rooted additive tree is called an *ultrametric tree* if the distances between any two leaves i and j and their common ancestor k are equal

$$d_{ik} = d_{jk}$$

- $d_{ij}/2$ corresponds to the time elapsed since divergence of i and j from the common parent
- In other words, edge lengths are measured by a *molecular clock* with a constant rate



Identifying ultrametric data

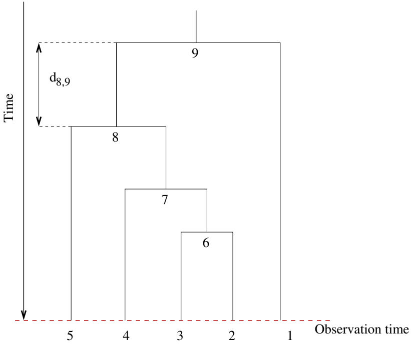
- We can identify distances to be ultrametric by the three-point condition:
  - D corresponds to an ultrametric tree if and only if for any three **species** we can label them i, j, and k such that the distances satisfy:

$$d_{ik} = d_{jk} \geq d_{ij}$$

- If we find out that the data is ultrametric, we can utilise a simple algorithm to find the corresponding tree

Ultrametric trees

- Only vertical segments of the tree have correspondence to some distance d_{ij}
- Horizontal segments act as connectors
- $d_{ik} = d_{jk}$ for any two leaves i, j and any ancestor k of i and j



UPGMA algorithm

- UPGMA (unweighted pair group method using arithmetic averages) constructs a phylogenetic tree via clustering
- The algorithm works by at the same time
  - Merging two clusters
  - Creating a new node on the tree
- The tree is built from leaves towards the root
- UPGMA produces a ultrametric tree

Cluster distances

- Let distance d_{ij} between clusters C_i and C_j be

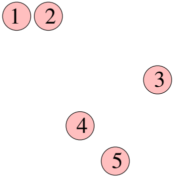
$$d_{ij} = \frac{1}{|C_i||C_j|} \sum_{p \in C_i, q \in C_j} d_{pq},$$

  that is, the average distance between points (species) in the cluster.

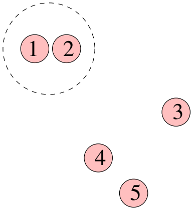
UPGMA algorithm

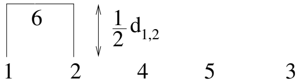
- Initialisation
  - Assign each point i to its own cluster C_i
  - Define one leaf for each point and place it at height zero
- Iteration
  - Find clusters i and j for which d_{ij} is minimal
  - Define new cluster k by $C_k = C_i \cup C_j$ and compute $d_{k\ell}$ for all ℓ
  - Add a node k with children i and j to the tree. Place k at height $d_{ij}/2$
  - Remove clusters i and j
- Termination
  - When only two clusters i and j remain, place root at height $d_{ij}/2$

UPGMA example

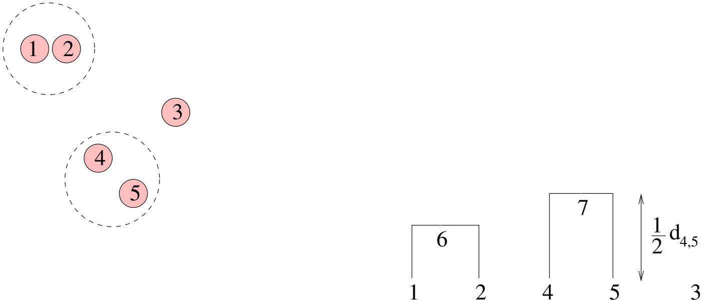


UPGMA example

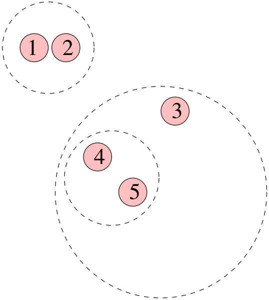


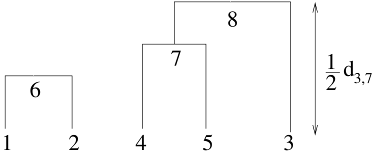


UPGMA example

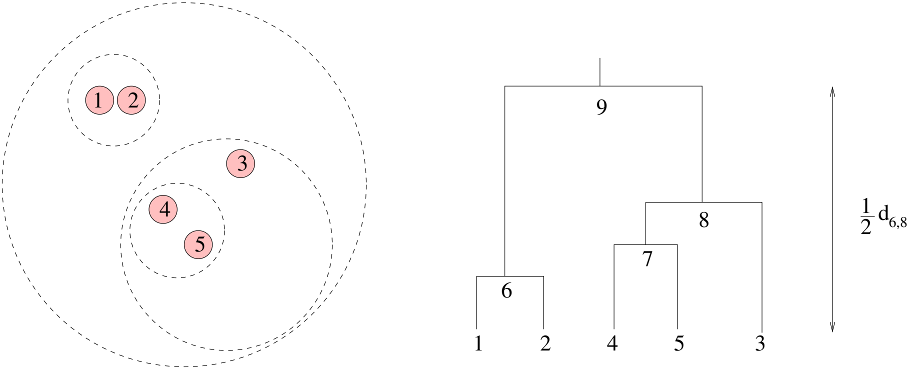


UPGMA example





UPGMA example



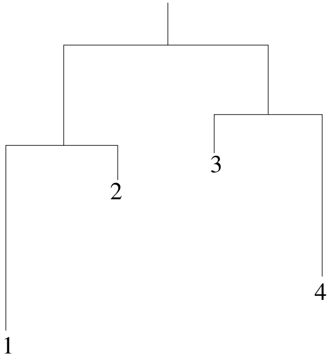
UPGMA implementation

- In naive implementation, each iteration takes $O(n^2)$ time with n initial points $\implies$ algorithm takes $O(n^3)$ time
- The algorithm can be implemented to take only $O(n^2)$ time (see Gronau & Moran, 2006, for a survey)

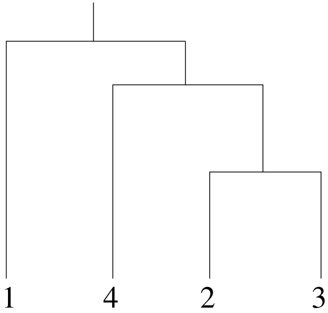
Problem solved?

- We now have a simple algorithm which finds an ultrametric tree
    - If the data is ultrametric, then there is exactly one ultrametric tree corresponding to the data
    - The tree found is then the “correct” solution to the phylogeny problem if the assumptions hold
- Unfortunately, the data is not ultrametric in practice
    - Measurement errors distort distances
    - Basic assumption of a molecular clock does not hold usually very well

Incorrect reconstruction of non-ultrametric data by UPGMA



Tree which corresponds to non-ultrametric distances



Incorrect ultrametric reconstruction by UPGMA algorithm

Outline

Distance-based clustering, UPGMA

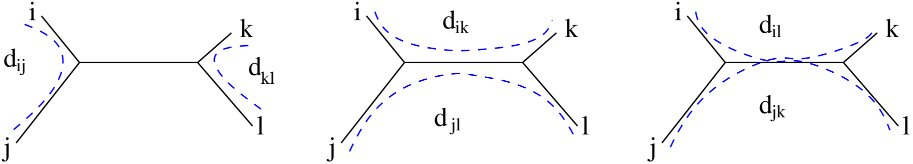
Neighbor joining

Study group assignments

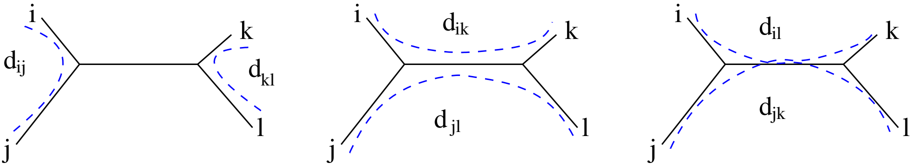
About the exam

Checking for additivity

- Recall: a tree is additive if for every pair of leaves the distance between the leaves is the sum of the edge distances on the path between the leaves.
- How can we check that the data is additive?
- Let i, j, k, and ℓ be four **distinct** species
- Compute three sums
  - $d_{ij} + d_{k\ell}$
  - $d_{ik} + d_{j\ell}$
  - $d_{i\ell} + d_{jk}$



Four-point condition



- Sums represented by the middle and right figures cover all edges
- Sum represented by the left figure does not cover all edges
- *Four-point condition*: i, j, k, and ℓ satisfy the four-point condition if two of the sums $d_{ij} + d_{k\ell}$, $d_{ik} + d_{j\ell}$, and $d_{i\ell} + d_{jk}$ are equal and the third one is smaller than these two.
- An $n \times n$ matrix D is additive if and only if the four-point condition holds for every 4 elements $1 \leq i, j, k, \ell \leq n$.

Checking for additivity: Example

	A	B	C	D
A	0	6	7	5
B		0	11	9
C			0	6
D				0

- $d_{AB} + d_{CD} = 6 + 6 = 12$
- $d_{AC} + d_{BD} = 7 + 9 = 16$
- $d_{AD} + d_{BC} = 5 + 11 = 16$

- Two of the sums are equal and the third is smaller

  $\Longrightarrow$ Four-point condition holds

  $\Longrightarrow$ Matrix is additive

Finding an additive phylogenetic tree

- Additive trees can be found for example by the neighbor joining method (Saitou & Nei, 1987)
- However, in real data, even additivity usually does not hold very well

Neighbor joining algorithm

- Neighbor joining works in a similar fashion to UPGMA
    - Find clusters C_1 and C_2 that minimize a function $f(C_1, C_2)$
    - Join the two clusters C_1 and C_2 into a new cluster C
    - Add a node to the tree corresponding to C
    - Assign distances to new branches
- Differences in
    - The choice of function $f(C_1, C_2)$
    - How to assign the distances

Neighbor joining algorithm: Separation of a cluster

- Let $u(C_i)$ be the *separation* of cluster C_i from other clusters defined as

$$u(C_i) = \frac{1}{n - 2} \sum_{C_j} d_{ij}$$

where n is the number of clusters.

Neighbor joining algorithm

- Neighbor joining at the same time
  - Minimizes the distance between clusters C_i and C_j to be joined
  - Maximizes the separation of both C_i and C_j from other clusters
- Recall that UPGMA only minimizes the distance between the clusters C_i and C_j

Neighbor joining algorithm

- Initialization as in UPGMA
- Iteration
  - Find clusters C_i and C_j for which $d_{ij} - u(C_i) - u(C_j)$ is minimum
  - Define a new cluster $C_k = C_i \cup C_j$ and compute $d_{k\ell}$ for all ℓ:

$$d_{k\ell} = \frac{1}{2}(d_{i\ell} + d_{j\ell} - d_{ij})$$

  - Remove clusters C_i and C_j
  - Define a node k with edges to i and j
  - Assign length $\frac{1}{2}(d_{ij} + u(C_i) - u(C_j))$ to the edge $i \to k$
  - Assign length $\frac{1}{2}(d_{ij} + u(C_j) - u(C_i))$ to the edge $j \to k$
- Termination
  - When two clusters i and j remain, add an edge between them with weight d_{ij}.

Neighbor joining algorithm: Example

	A	B	C	D
A	0	6	7	5
B		0	11	9
C			0	6
D				0

i	$u(C_i)$
A	$(6+7+5)/2 = 9$
B	$(6+11+9)/2 = 13$
C	$(7+11+6)/2 = 12$
D	$(5+9+6)/2 = 10$

i, j	$d_{ij} -$	$u(C_i) -$	$u(C_j)$	
A,B	$6 -$	$9 \quad -$	13	$= -16$
A,C	$7 -$	$9 \quad -$	12	$= -14$
A,D	$5 -$	$9 \quad -$	10	$= -14$
B,C	$11 -$	$13 \quad -$	12	$= -14$
B,D	$9 -$	$13 \quad -$	10	$= -14$
C,D	$6 -$	$12 \quad -$	10	$= -16$

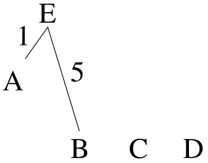
Choose either one of the red pairs to join

Neighbor joining algorithm: Example

	A	B	C	D
A	0	6	7	5
B		0	11	9
C			0	6
D				0

i	$u(C_i)$
A	$(6+7+5)/2=9$
B	$(6+11+9)/2=13$
C	$(7+11+6)/2=12$
D	$(5+9+6)/2=10$

i,j	$d_{ij}-u(C_i)-\ u(C_j)$
A,B	$6-\ 9\ -\ 13\ =-16$
A,C	$7-\ 9\ -\ 12\ =-14$
A,D	$5-\ 9\ -\ 10\ =-14$
B,C	$11-\ 13\ -\ 12\ =-14$
B,D	$9-\ 13\ -\ 10\ =-14$
C,D	$6-\ 12\ -\ 10\ =-16$



$$d_{AE}=\tfrac{1}{2}6+\tfrac{1}{2}(9-13)=1$$
$$d_{BE}=\tfrac{1}{2}6+\tfrac{1}{2}(13-9)=5$$

This is only the first step!

Neighbor joining algorithm: Correctness

- **Theorem:** If D is an additive matrix, neighbor joining algorithm correctly constructs the corresponding additive tree.
    - *Proof (sketch):* By contradiction. Assume i and j with minimum $D_{ij} = d_{ij} - u(C_i) - u(C_j)$ are not neighbors in the additive tree. Show that there are two neighbors m and n with $D_{mn} < D_{ij}$ (see Durbin et al. *Biological Sequence Analysis*, pp. 190-191 for details). Then the theorem follows by induction.

Outline

Distance-based clustering, UPGMA

Neighbor joining

Study group assignments

About the exam

Study Group 1: Those who did not get any material at lecture

- Read pages 179–181 from Sung: Algorithms in Bioinformatics, CRC Press, 2010
  - The 4-point condition for additive trees.
  - Copies distributed at lecture.
- At study group, explain the proof of Theorem 7.1 (Buneman's 4-point condition).
- The *last* part of the proof is very condensed:
  - Why does the path length between a and b in T equal $M_{ab} + M_{bc} - M_{cd}$? (recall that T' and T'' are additive trees)
  - The 4-point condition can be applied because $M_{ad} + M_{bc} \geq M_{ac} + M_{bd}$. Why is this true? (Recall how c and d were chosen)

Study Group 2: Random allocation at lecture

- Read pages 184–187 from Sung: Algorithms in Bioinformatics, CRC Press, 2010 (Especially Lemma 7.13).
  - Correctness of UPGMA algorithm
  - Copies distributed at lecture.
- At study group, summarize the proof for the correctness of UPGMA.

Study Group 3: Random allocation at lecture

- Read pages 190–191 from Durbin et al.: Biological Sequence Analysis, Cambridge University Press, 1998.
    - Correctness of neighbor joining.
    - Note that their notation of D_{ij} equals our $d_{ij} - u(C_i) - u(C_j)$.
    - Copies distributed at lecture.
- At study group, summarize the proof for correctness of neighbor joining.

Outline

Distance-based clustering, UPGMA

Neighbor joining

Study group assignments

About the exam

Practicalities

- The course exam is on Wed 16.10. at 16:00 in hall A111
  - 2.5 hours time
  - You can leave at earliest half an hour after the start of the exam
- The first separate exam is on Tue 26.11. at 16:00 in hall B123
  - This can also be taken as renewal exam where points from exercises are still valid!
  - 3.5 hours time
  - this will probably be graded in a month's time
- No own papers.
- You will need student id card (or some other proof of identity)!
- You can answer in English or Finnish.

What to study for the exam?

- Material covered at the lectures!
- Take a look at some subjects studied in the study groups. If there are questions regarding subjects in the study groups, you will have a choice so that you can answer to a question about a subject you have studied yourself.
    - Example: Choose one of the (non-trivial) problems studied during the course (in study groups, lectures, or/and exercises) not related to the previous assignments above. Define the problem (input, output), explain how the problem is motivated by molecular biology, and describe an algorithm for the problem either simulating an example or by giving its pseudocode.

What kind of questions?

- In course exam four questions, some might include subquestions (i.e. several questions that all require short answers)
- In separate exams five questions.
- Short answers:
  - Example: Explain in one or two sentences what is the shortest superstring problem.
- Essay type questions:
  - Example: Define the Motif Finding and Median String problems and explain why they are actually the same problem. Describe briefly the idea of the branch-and-bound solution for solving the problem.
- Simulate an algorithm on a given input.
- Solve an exercise.