Bioinformatics Algorithms

(Fundamental Algorithms, module 2)

Zsuzsanna Lipták

Masters in Medical Bioinformatics
academic year 2018/19, II semester

Phylogenetics II[1]

[1]These slides are partially based on the Lecture Notes from Bielefeld University "Algorithms for Phylogenetic Reconstruction" (2016/17), by J. Stoye, R. Wittler, et al.

Character data

Now the input data consists of states of characters for the given objects, e.g.

- morphological data, e.g. number of toes, reproductive method, type of hip bone, . . . or
- molecular data, e.g. what is the nucletoide in a certain position.

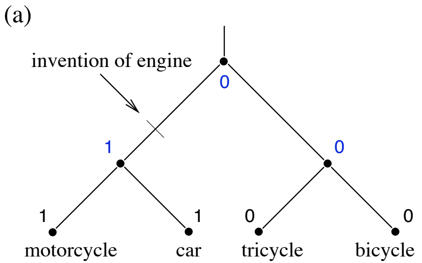
Character data

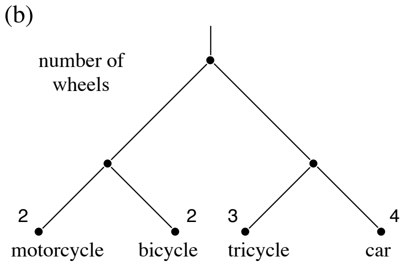
Example

	C_1 : # wheels	C_2 : existence of engine
bicycle	2	0
motorcycle	2	1
car	4	1
tricycle	3	0

- objects (species): Bicycle, motorcycle, tricycle, car
- characters: number of wheels; existence of an engine
- character states: $2, 3, 4$ for C_1;
  $0, 1$ for C_2 ($1 =$ YES, $0 =$ NO)
- This matrix M is called a character-state-matrix, of dimension $(n \times m)$, where for $1 \leq i \leq n, 1 \leq j \leq m$: $M_{ij} =$ state of character j for object i. (Here: $n = 4, m = 2$.)

Character data





Two different phylogenetic trees for the same set of objects.

Character data

We want to avoid

- parallel evolution (= convergence)
- reversals

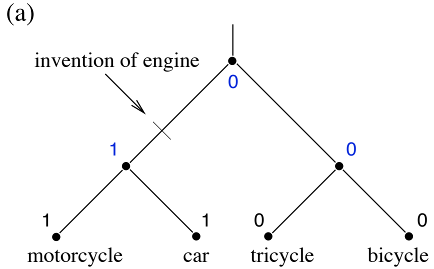
Together these two conditions are also called homoplasies.

Mathematical formulation: compatibility.

Compatibility

Definition

A character is compatible with a tree if all inner nodes of the tree can be labeled such that each character state induces one connected subtree.

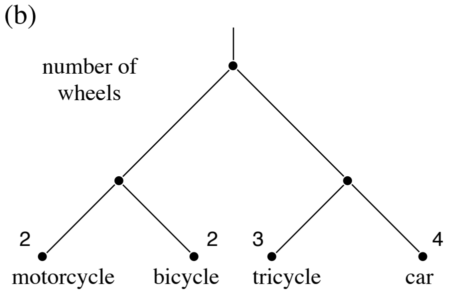


This tree is compatible with C_2, one possibility of labeling the inner nodes is shown.

Compatibility

Definition

A character is compatible with a tree if all inner nodes of the tree can be labeled such that each character state induces one connected subtree.

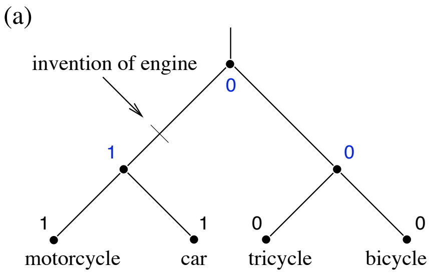


This tree is compatible with C_1. (We have to give a labeling of the inner nodes to prove this.) It is not compatible with C_2 (why?)

Compatibility

Definition

A character is compatible with a tree if all inner nodes of the tree can be labeled such that each character state induces one subtree (i.e. is connected).



This tree is also compatible with C_1: We have to give a labeling of the inner nodes (w.r.t. C_1) to prove this. (Exercise!)

Compatibility

Here is another example input character-state matrix (here $n = 5, m = 2$):

	C_1	C_2
α	A	A
β	A	C
γ	C	C
δ	C	G
ϵ	G	G

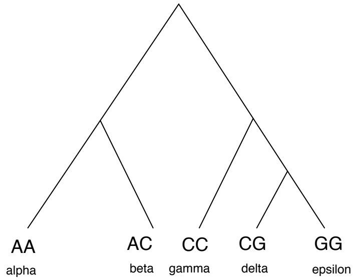
Our goal is to find a tree that is compatible with every character. Such a tree is called Perfect Phylogeny.

Perfect Phylogeny

Definition

A tree T is called a perfect phylogeny (PP) for $\mathcal{C}$ if all characters $C \in \mathcal{C}$ are compatible with T.

Example



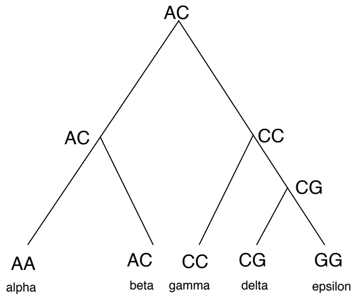
Why? We have to find a labeling of the inner nodes s.t. for both characters C_1 and C_2, each state induces a subtree.

Perfect Phylogeny

Definition

A tree T is called a perfect phylogeny (PP) for the character-state matrix M if all characters are compatible with T.

Example



Note: Our tree (b) for the vehicles was also a PP, since it is compatible both with C_1 and with C_2.

Perfect Phylogeny

Theorem

Let M be a character-state matrix of dimension $n \times m$, and for $1 \leq i \leq m$, let $r_i =$ number of distinct states in column i (i.e. the number of states which actually occur). Then a tree T is a perfect phylogeny (PP) for M if and only if $pc(T) = \sum_{i=1}^{m}(r_i - 1)$.

Example

For the previous example, we have $r_1 = r_2 = 3$, so a tree T is a PP iff $pc(T) = 2 + 2 = 4$.

Example

For the vehicle-example, we have $r_1 = 2, r_2 = 3$, therefore if $pc(T) = 3$, then a tree is a PP.

Perfect Phylogeny

- Ideally, we would like to find a PP for our input data.

Perfect Phylogeny

- Ideally, we would like to find a PP for our input data.
- Deciding in general whether a PP exists is NP-hard.
  (More precisely: For characters with number of states ≥ 4, the PP problem is NP-hard.)

Perfect Phylogeny

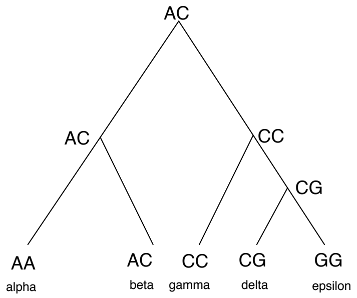
- Ideally, we would like to find a PP for our input data.
- Deciding in general whether a PP exists is NP-hard.
  (More precisely: For characters with number of states ≥ 4, the PP problem is NP-hard.)
- Doesn't really matter, since most of the time, no PP exists anyway.
  Why: due to homoplasies; because our input data has errors; our evolutionary model may not be adequate; and, and, and . . .

Perfect Phylogeny

- Ideally, we would like to find a PP for our input data.
- Deciding in general whether a PP exists is NP-hard.
  (More precisely: For characters with number of states ≥ 4, the PP problem is NP-hard.)
- Doesn't really matter, since most of the time, no PP exists anyway.
  Why: due to homoplasies; because our input data has errors; our evolutionary model may not be adequate; and, and, and . . .
- Therefore we usually want to find a best possible tree.

Parsimony

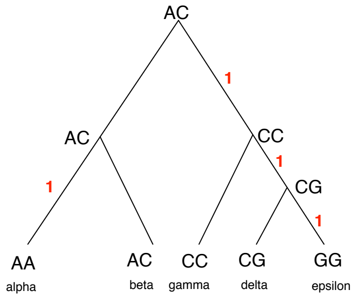
Parsimony: What is a best possible tree?



Why is this tree "perfect"?

Parsimony

What is a best possible tree?



Why is this tree "perfect"?

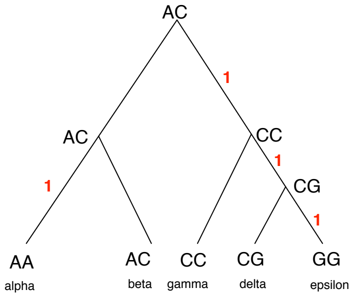
Because it has few changes of states!

In red, we marked the edges where there are state changes (an evolutionary event happened), and how many (in this case, always 1).

Parsimony

Definition

The parsimony cost of a phylogenetic tree with labeled inner nodes is the number of state changes along the edges (i.e. the sum of the edge costs, where the cost of an edge = number of characters whose state differs between child and parent).

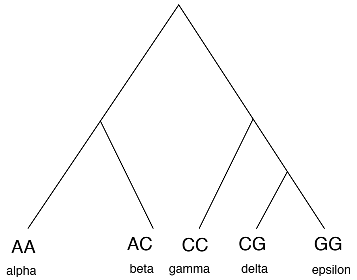


The parsimony cost of this labeled tree is 4.

Parsimony

Definition

The parsimony cost of a phylogenetic tree (without labels on the inner nodes) is the minimum of the parsimony cost over all possible labelings of the inner nodes.



The parsimony cost of this tree is 4, because the best labeling has cost 4.

Parsimony

Phylogenetic Reconstruction with Character Data

Given a character-state matrix M, our goal is to find a phylogenetic tree which minimizes the parsimony cost.

We split the problem into two sub-problems:

1. Small Parsimony: Given a phylogenetic tree, find its parsimony cost, i.e. find a most parsimonious labeling of the inner nodes. This problem can be solved efficiently.
2. Large Parsimony or Maximum Parsimony: Find a tree with minimum parsimony cost. This problem is NP-hard.

Small Parsimony

Small Parsimony Problem

Given: a phylogenetic tree T with character-states at the nodes.
Find: a labeling of the inner nodes with states with minimum parsimony cost.

Algorithm

This problem can be solved using Fitch' algorithm, which runs in time $O(nmr)$, where $n =$ number of species, $m =$ number of characters, and $r =$ maximum number of states over all characters.

Maximum Parsimony

Maximum Parsimony Problem

The maximum parsimony problem is, given a character-state matrix, find a phylogenetic tree with lowest parsimony cost (= a "most parsimonious tree").

- When a PP exists, then it is also the most parsimonious tree.
- In general, the Maximum Parsimony Problem is NP-hard.

Summary for character data

- When the input is a character-state matrix, then we would like to find a tree which is compatible with each character.

Summary for character data

- When the input is a character-state matrix, then we would like to find a tree which is compatible with each character.
- Such a tree is called a perfect phylogeny (PP).

Summary for character data

- When the input is a character-state matrix, then we would like to find a tree which is compatible with each character.
- Such a tree is called a perfect phylogeny (PP).
- PPP is NP-hard (for number of states ≥ 4).
- Usually, no PP exists, therefore in general ...

Summary for character data

- When the input is a character-state matrix, then we would like to find a tree which is compatible with each character.
- Such a tree is called a perfect phylogeny (PP).
- PPP is NP-hard (for number of states ≥ 4).
- Usually, no PP exists, therefore in general ...
- We are looking for a most parsimonious tree (a tree with lowest parsimony cost).

Summary for character data

- When the input is a character-state matrix, then we would like to find a tree which is compatible with each character.
- Such a tree is called a perfect phylogeny (PP).
- PPP is NP-hard (for number of states ≥ 4).
- Usually, no PP exists, therefore in general ...
- We are looking for a most parsimonious tree (a tree with lowest parsimony cost).
- The parsimony cost is defined as the minimum number of the state changes on the edges over all possible labelings of the inner nodes.

Summary for character data

- When the input is a character-state matrix, then we would like to find a tree which is compatible with each character.
- Such a tree is called a perfect phylogeny (PP).
- PPP is NP-hard (for number of states ≥ 4).
- Usually, no PP exists, therefore in general ...
- We are looking for a most parsimonious tree (a tree with lowest parsimony cost).
- The parsimony cost is defined as the minimum number of the state changes on the edges over all possible labelings of the inner nodes.
- Recall: There are super-exponentially many trees on n taxa (both rooted and unrooted), so we cannot try them all.

Summary for character data (cont'ed)

- The problem of finding a most parsimonious tree (a tree with lowest parsimony cost) is split into Small Parsimony and Maximum Parsimony:

Summary for character data (cont'ed)

- The problem of finding a most parsimonious tree (a tree with lowest parsimony cost) is split into Small Parsimony and Maximum Parsimony:
- Small Parsimony can be solved efficienly, e.g. by Fitch' algorithm.

Summary for character data (cont'ed)

- The problem of finding a most parsimonious tree (a tree with lowest parsimony cost) is split into Small Parsimony and Maximum Parsimony:
- Small Parsimony can be solved efficienly, e.g. by Fitch' algorithm.
- Maximum Parsimony is NP-hard, so probably no efficient algorithms exist.