Algorithms in Comparative Genomics

Universität Bielefeld, SS 2024
Dr. Marília D. V. Braga · Dr. Roland Wittler · M.Sc. Leonard Bohnenkämper
https://gi.cebitec.uni-bielefeld.de/teaching/2024summer/cg

Exercise sheet 3, 26.04.2024

Exercise 1 (Small Parsimony under SCJ—Sparsity) (5 pts)

Consider the phylogenetic tree $((\mathbb{A}, \mathbb{B}), (\mathbb{C}, \mathbb{D}))$; with the following genomes at its leaves. Perform the bottom-up phase of the Fitch Algorithm for each adjacency (presence/absence). Only consider adjacencies that occur at any leaf.

$$\begin{aligned} \mathbb{A} &= [1\ 2\ 3\ 4] \\ \mathbb{B} &= [1\ 3\ 2\ 4] \\ \mathbb{C} &= [1\ 2\ 3\ 4] \\ \mathbb{D} &= [2\ 1\ 3\ 4] \end{aligned}$$

1. Would initializing the root with the presence of an adjacency in case of ambiguity result in consistent sets of adjacencies?
2. How many adjacencies would be reconstructed, if instead the absence of an adjacency is preferred? What is the total cost of the scenario?

Exercise 2 (Small Parsimony under SCJ—Exhaustiveness) (5 pts)

The Fitch-Algorithm as presented in the lecture is not guaranteed to find all optimal solutions.

1. Provide an example for the above statement.
2. Could your example still be found if the restriction to select 0 for the root node in case of ambiguity is lifted? If not: can you find such an example?
3. Does any of the missed solutions in your latter example contain fewer 0s than any of the found solutions? If not: can you find such an example?
4. Phrase an algorithm / a modification of the Fitch algorithm that is guaranteed to always find all solutions.

Exercise 3 (Big Parsimony under SCJ—NP-hardness) (5 pts)

As discussed in the lecture, the Big Parsimony Problem under SCJ is NP-hard. Feijão and Meidanis (TCBB 2011) show this by reduction from a Steiner Tree Problem in $\{0, 1\}^N$.

Here, we consider the general Steiner Tree Problem – one of the famous Karp's 21 NP-complete problems. The decision version can be phrased as follows: "Given an undirected graph and a subset of vertices, referred to as *terminals*, is there a subtree of size at most k that includes all terminals?" .

Research an NP-hardness proof and understand it.