Bielefeld University
Faculty of Technology

AG Genominformatik
Dr. Marília D. V. Braga
Dr. Daniel Dörr

Algorithms in Genome Research
Winter 2019/2020

Exercises

Number 6, Discussion: 2019 December 20

1. The set of compatible characters defines an inclusion tree. To see this, we introduce the *object set* (in the lecture referred to as *taxa*, i.e., the row labels of M) of a character: Given a compatible character matrix M, a character c corresponds to the set of *objects* O_c where the character c in its respective column is "on", that is it has value 1. Then the following holds true: *For any two characters c,d of M, either $O_c \subseteq O_d$, or $O_d \subseteq O_c$, or $O_c \cap O_d = \emptyset$.*

   (a) In testing a character matrix for compatibility, how can this insight be exploited to develop a faster algorithm that does not do all pairwise comparisons of columns?

   (b) The perfect phylogeny is a refined and annotated version of the inclusion tree. To ensure that the inclusion tree is in fact a *tree* and not a forest, we always assume a *root* object set O_r composed of all objects of the character matrix. The perfect phylogeny is then constructed as follows:

      1. Annotate all leaf nodes labeled c with the object set O_c, mark these objects as used, and annotate the parent edge with c.
      2. In a bottom-up fashion, for each internal node which is labeled with character c and whose children are all re-labeled, do
         i. Re-label the node with all objects of O_c which are not yet marked as 'used',
         ii. mark these objects as 'used', and
         iii. annotate the parent edge with c (not for the root node).
      3. For each node u and each object o it is labeled with, if u is not a leaf labeled only with o, remove o from the labeling and append a new leaf v labeled with o.

   Reconstruct the perfect phylogeny of the following character matrix:

$$M = \begin{pmatrix} 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 \\ 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{pmatrix}$$

2. An LP-relaxation of an ILP is the linear program without the integer constraints of its variables. Let P be an ILP and P' is its LP relaxation. Suppose that there is an optimal solution to P' where all of the variables take on integer values. Is the optimal solution of P' also an optimal solution to P?

3. In the lecture, we discussed an ILP for finding a largest independent set in a graph. Construct an ILP that finds two largest independent sets in a graph.