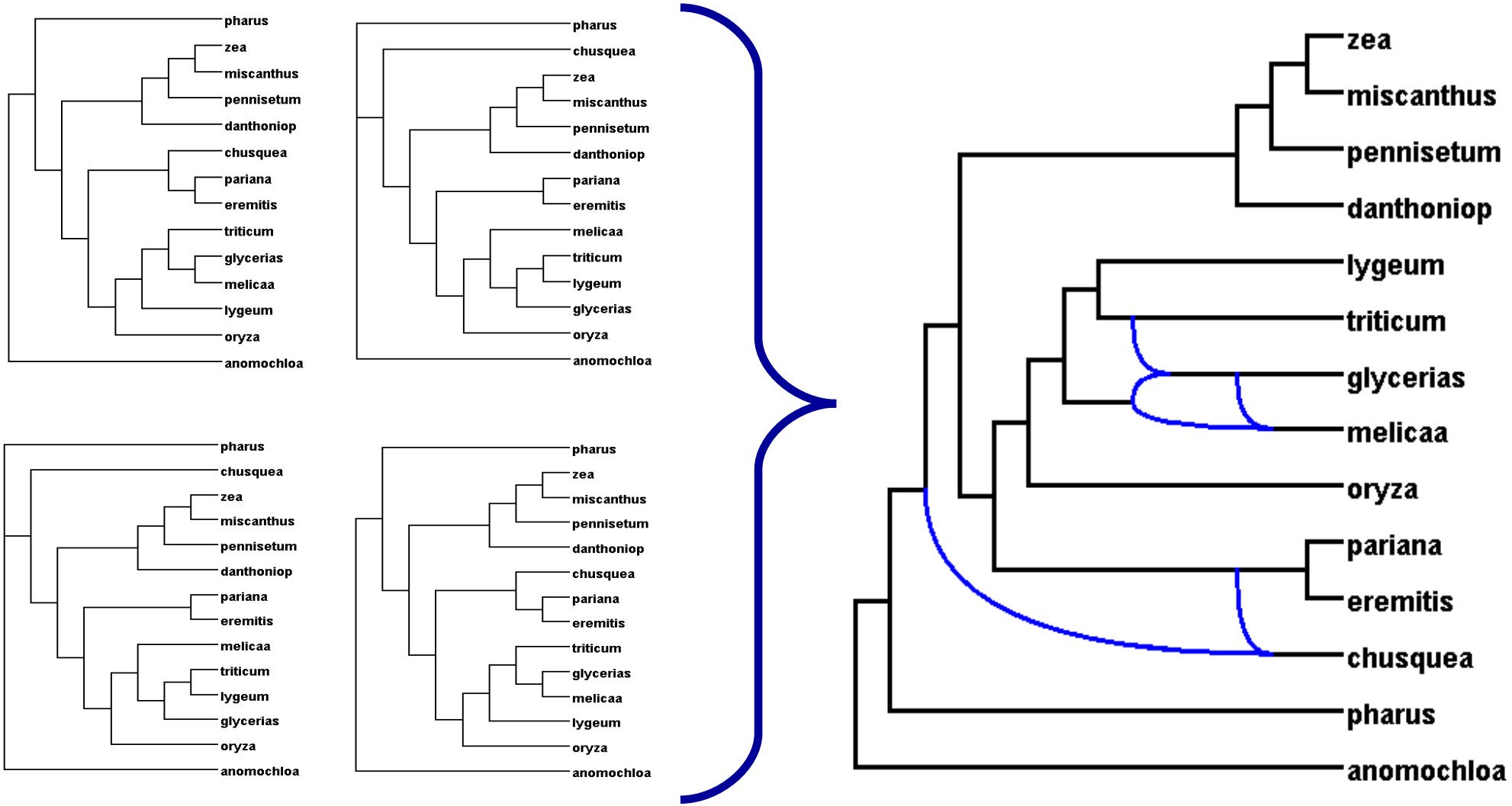
Phylogenetic Networks

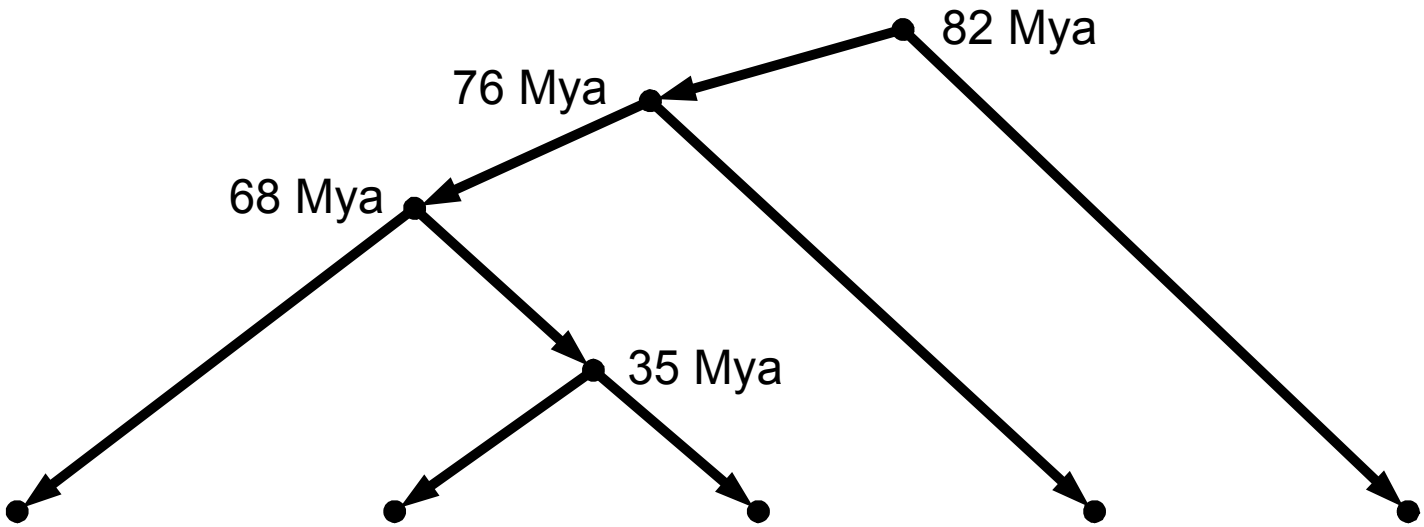
Reconstructing evolution

Leo van Iersel

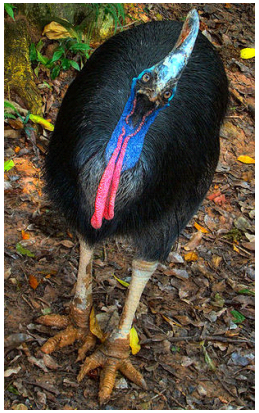
Algorithms for Combining Phylogenetic Trees into a Network



Phylogenetic Tree

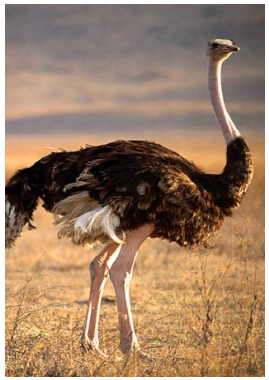


Kiwi
(New Zealand)

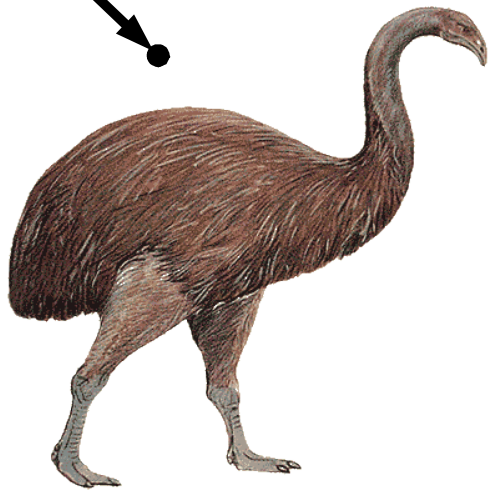


Cassowary
(New Guinea + Australia)

Emu
(Australia)

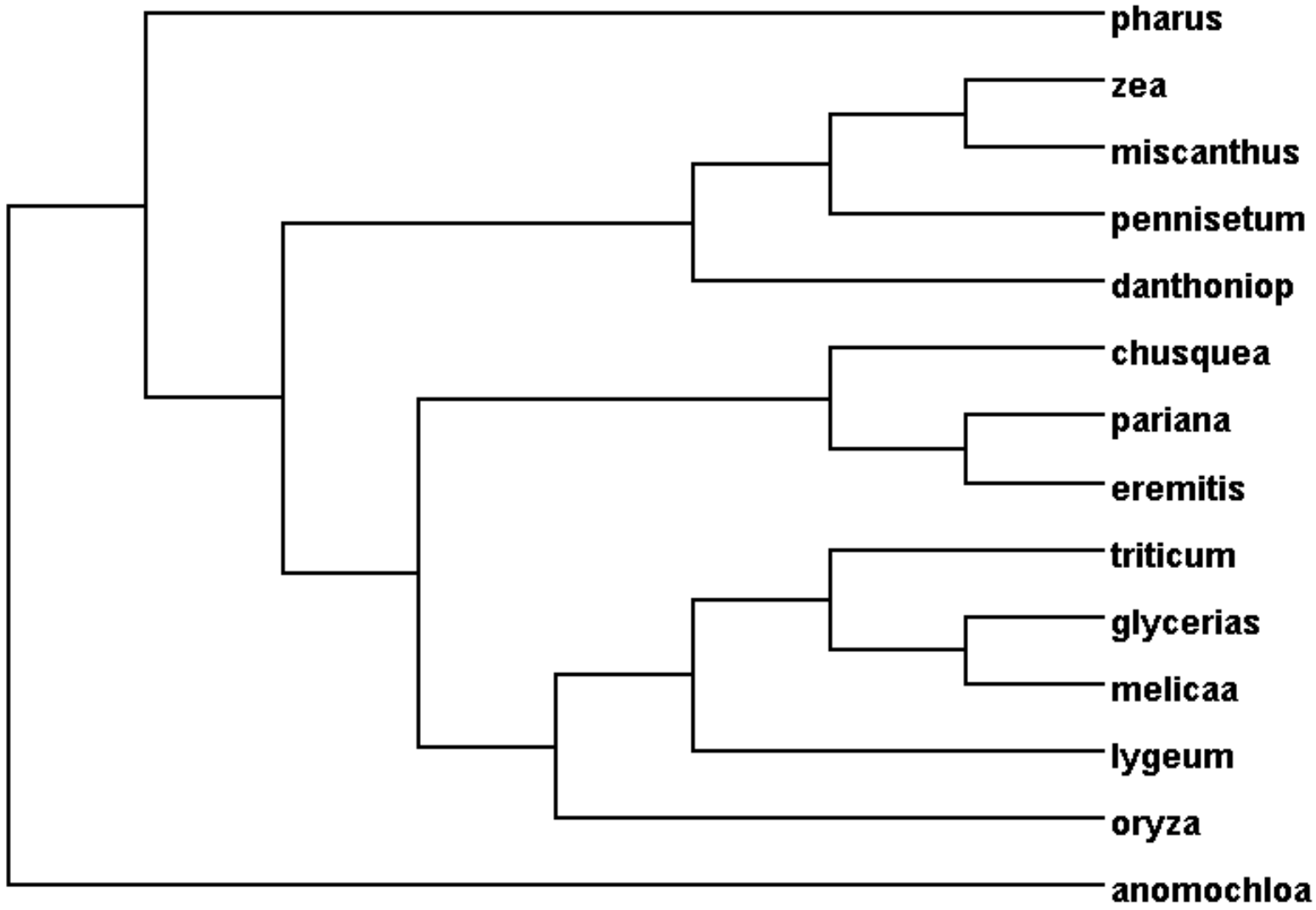


Ostrich
(Africa)

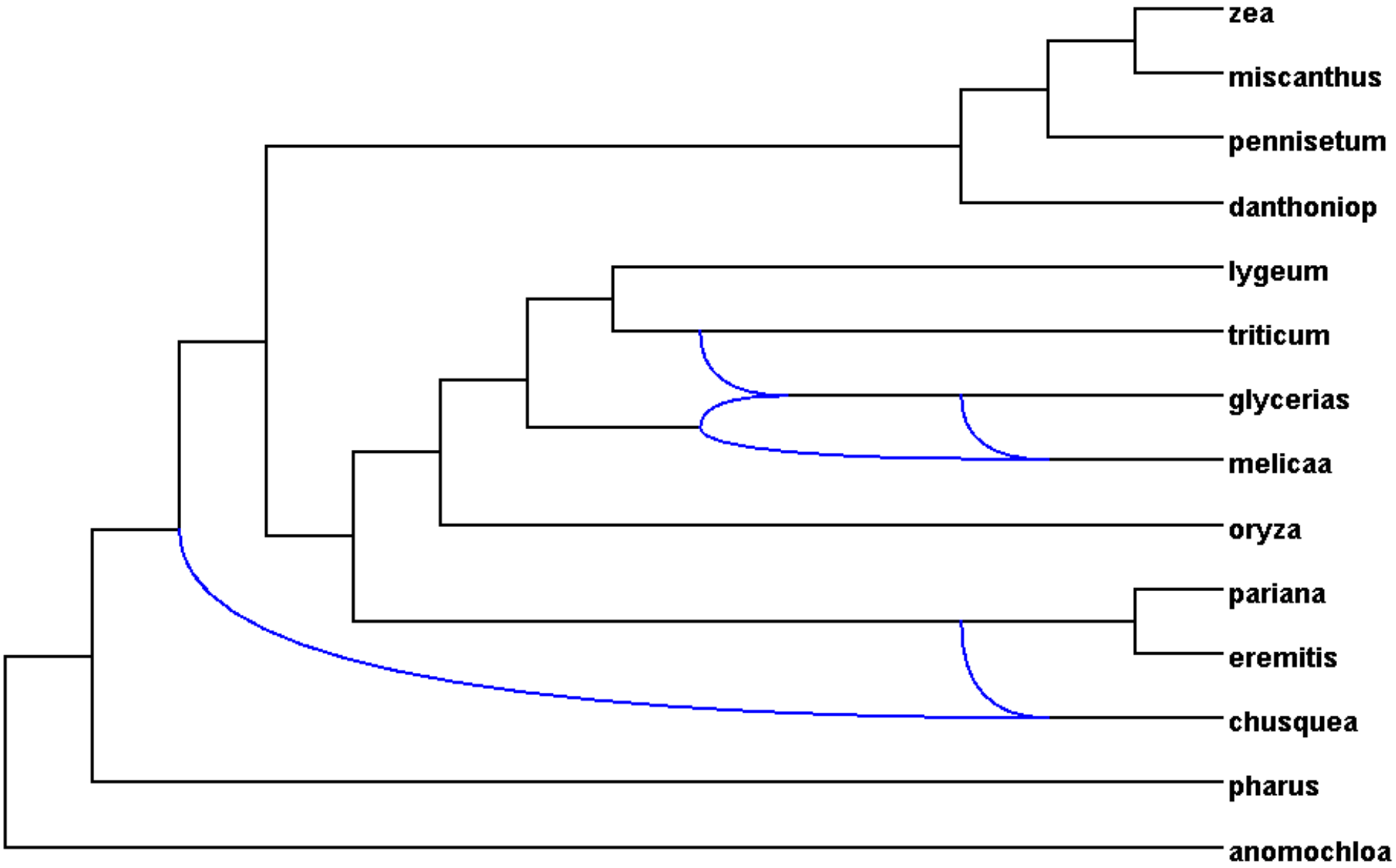


Moa
(New Zealand)

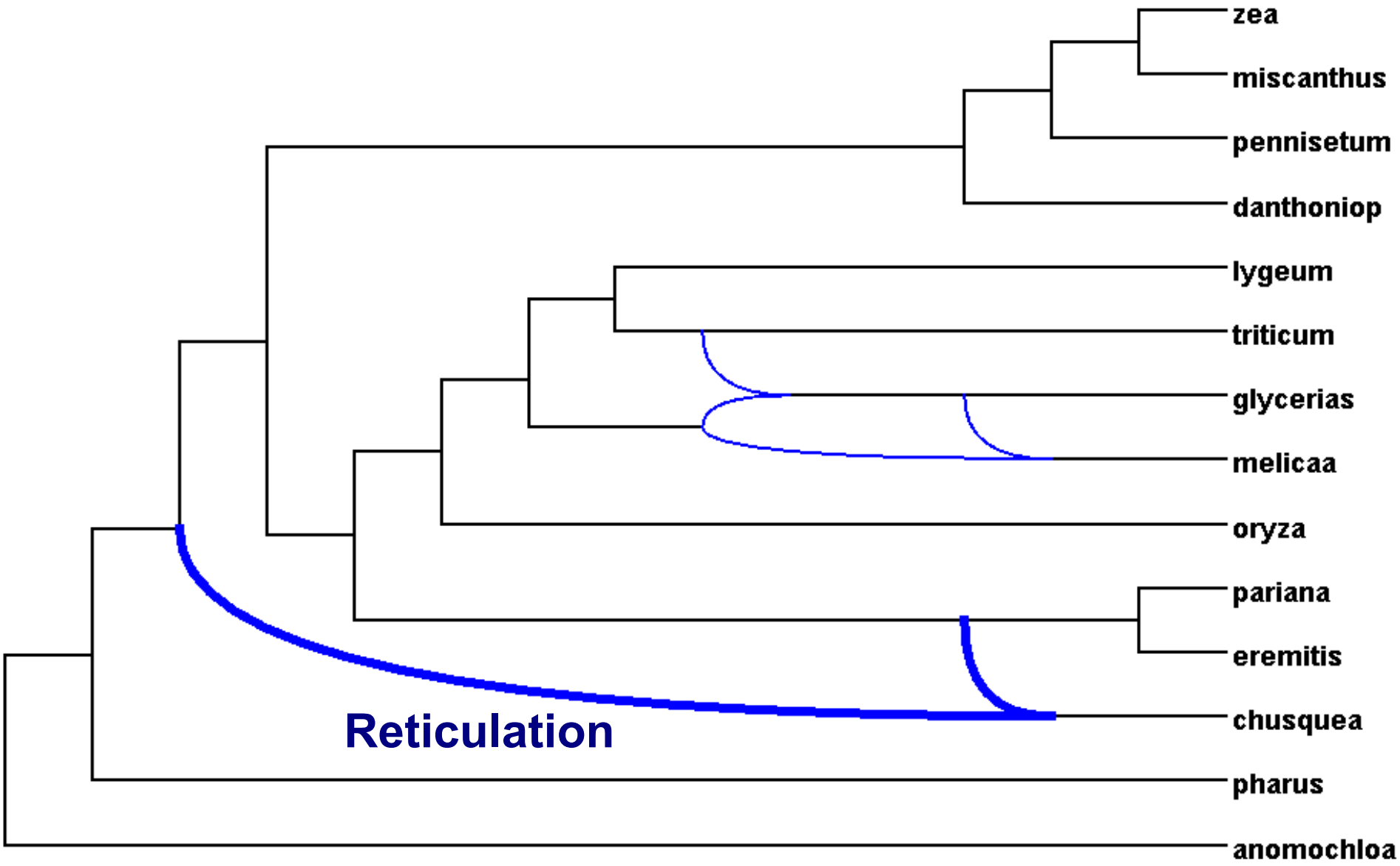
Phylogenetic Tree

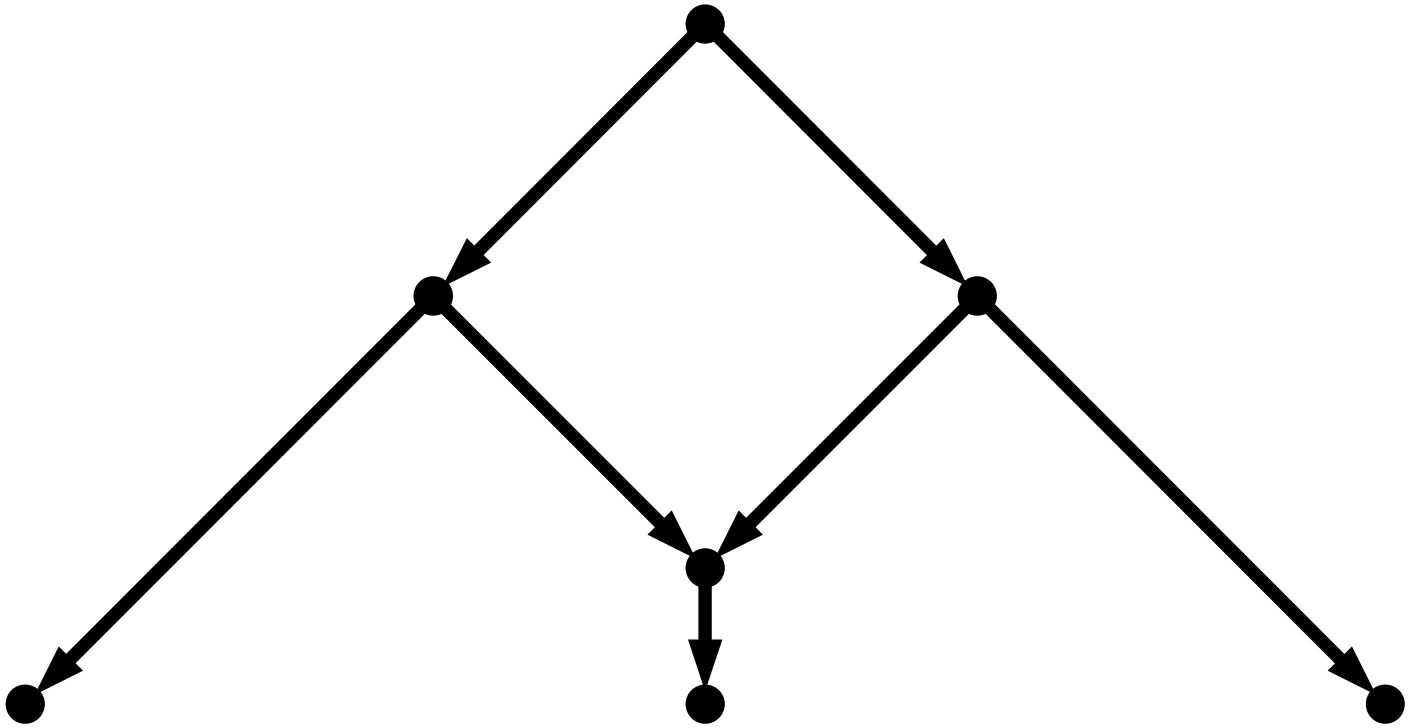


Phylogenetic Network



Phylogenetic Network

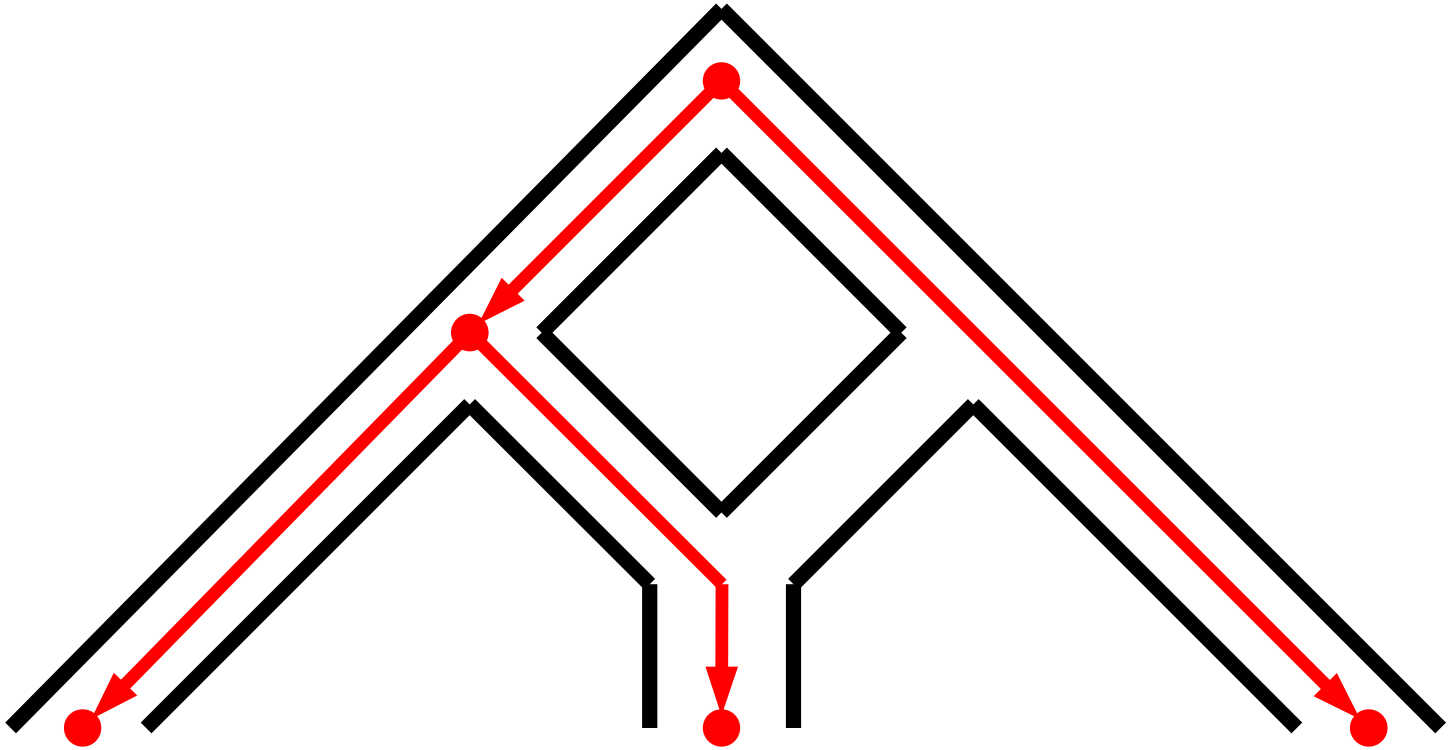




Parasitic Jaeger

Pomarine Skua

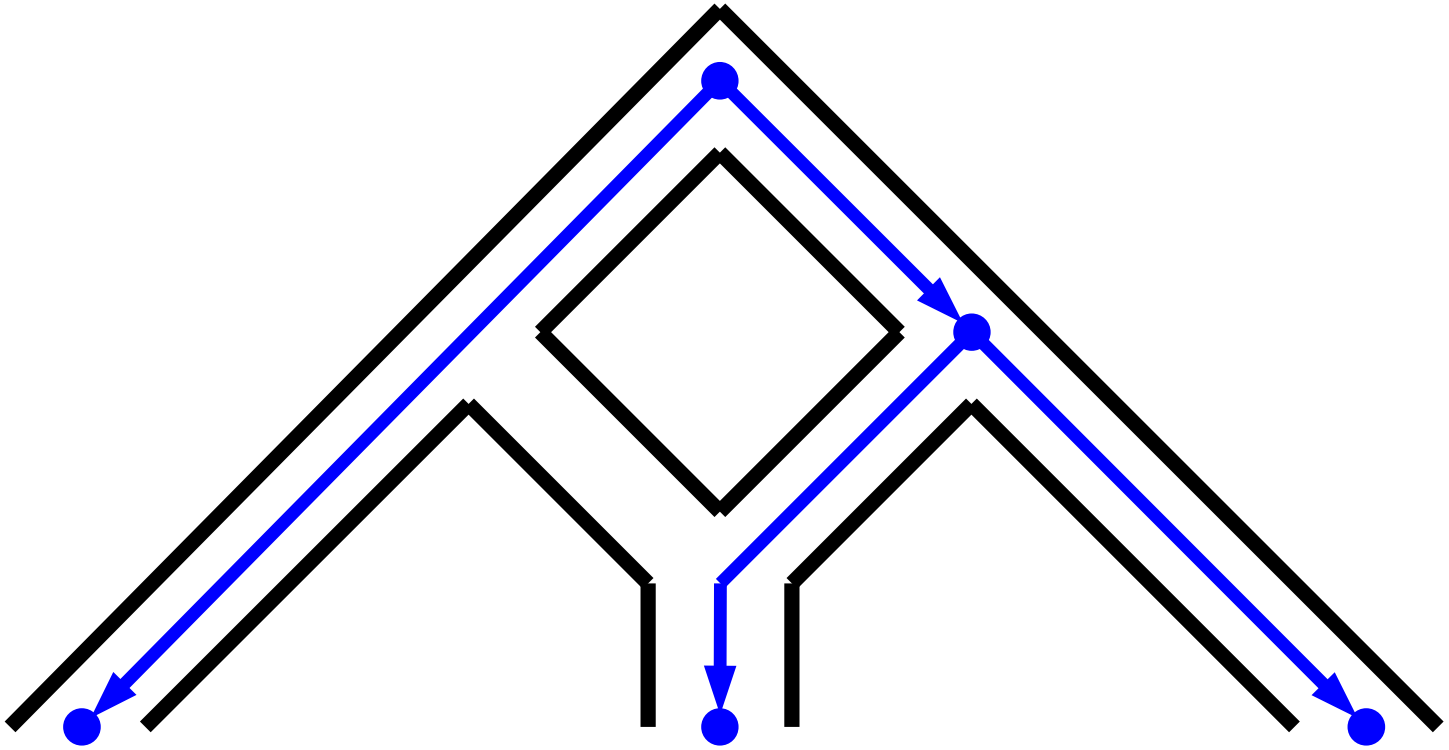
Great Skua



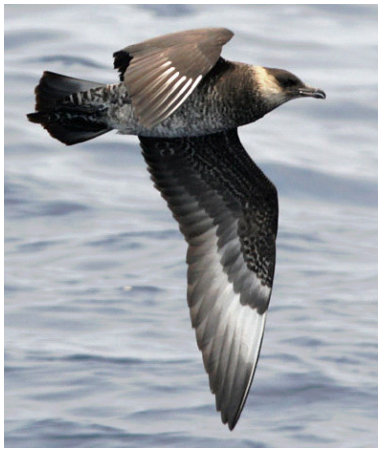
Parasitic Jaeger

Pomarine Skua

Great Skua



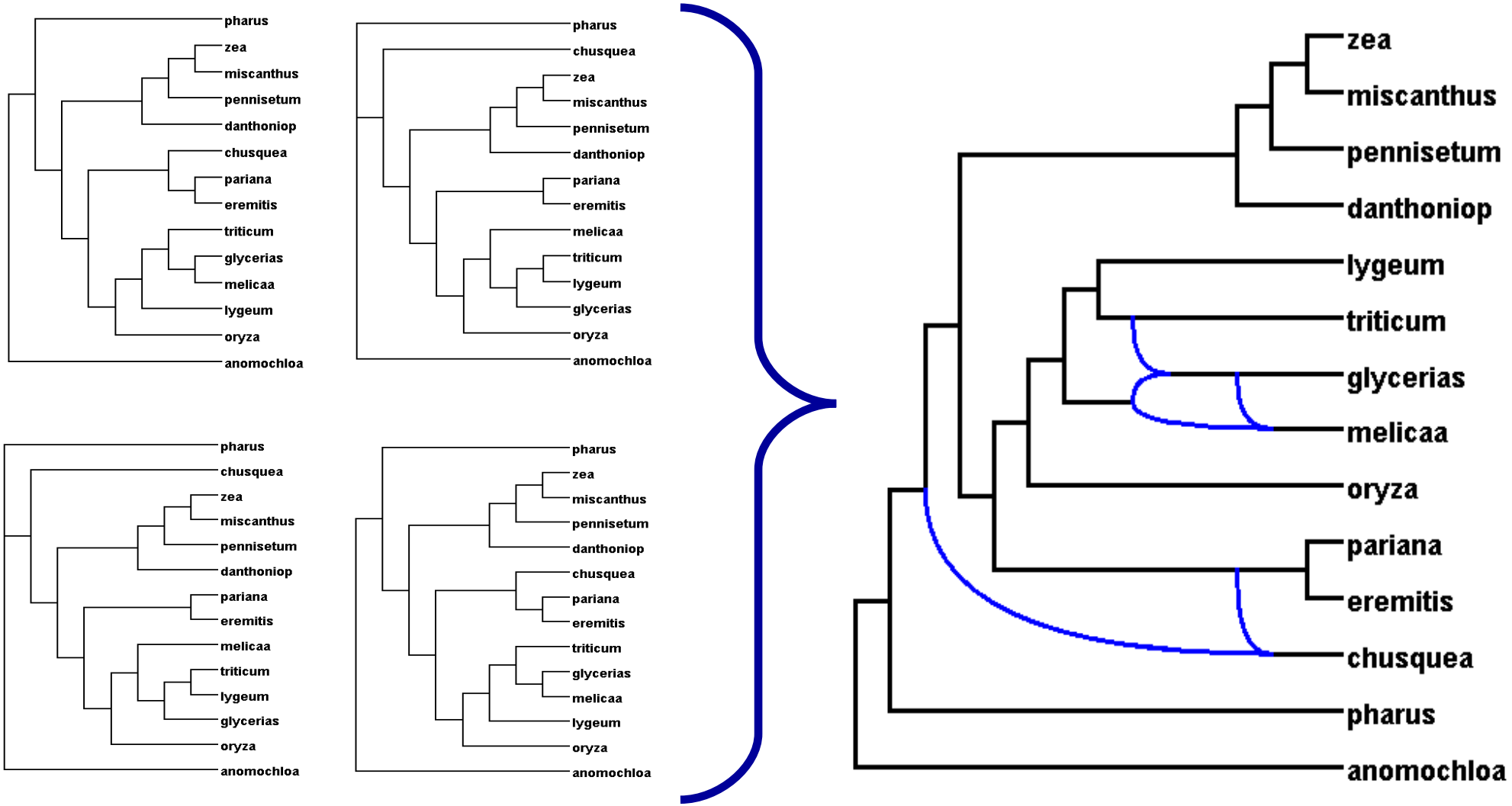
Parasitic Jaeger



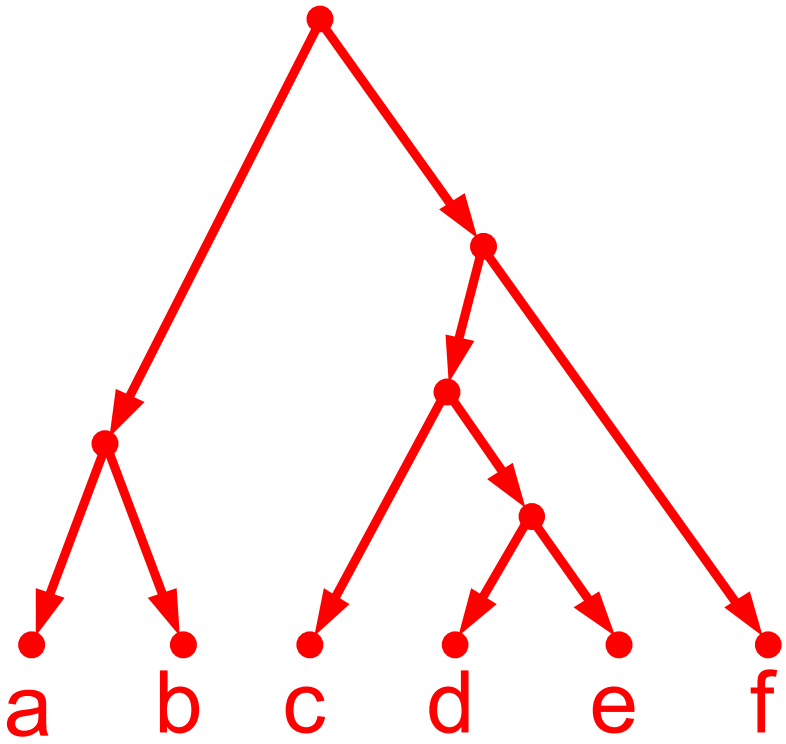
Pomarine Skua

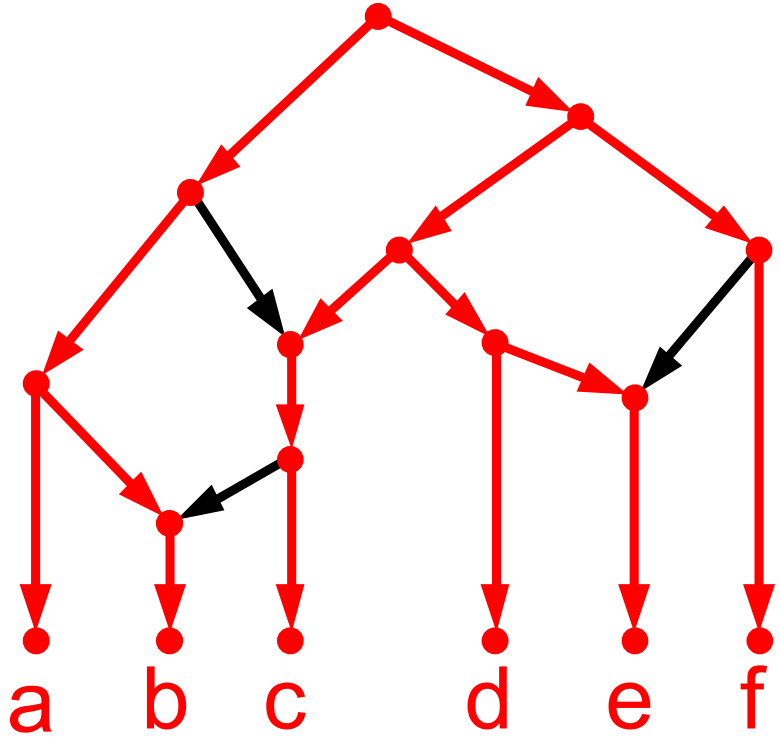
Great Skua

Combining Phylogenetic Trees into a Phylogenetic Network

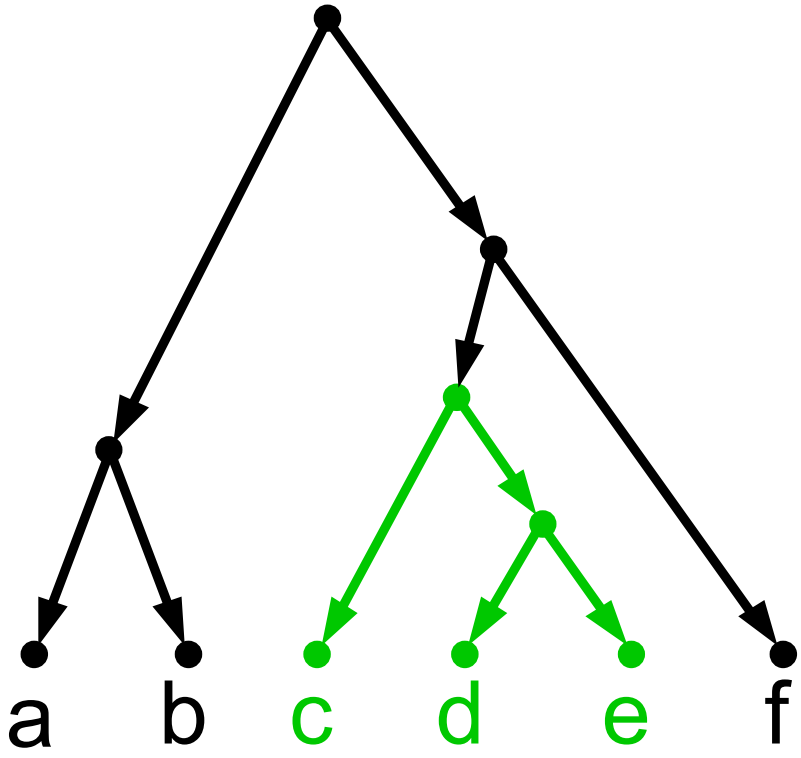


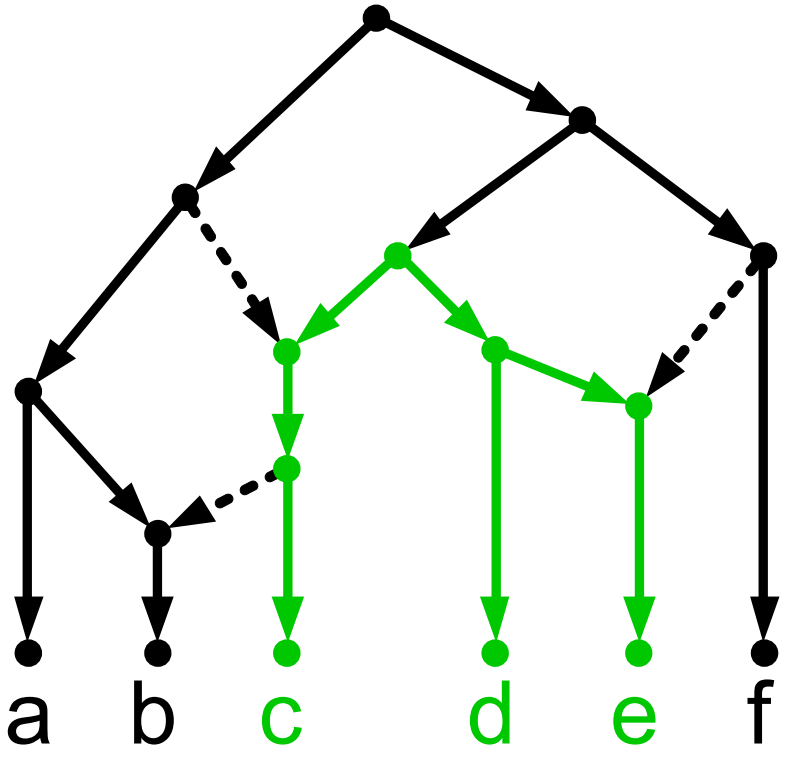
Tree Model



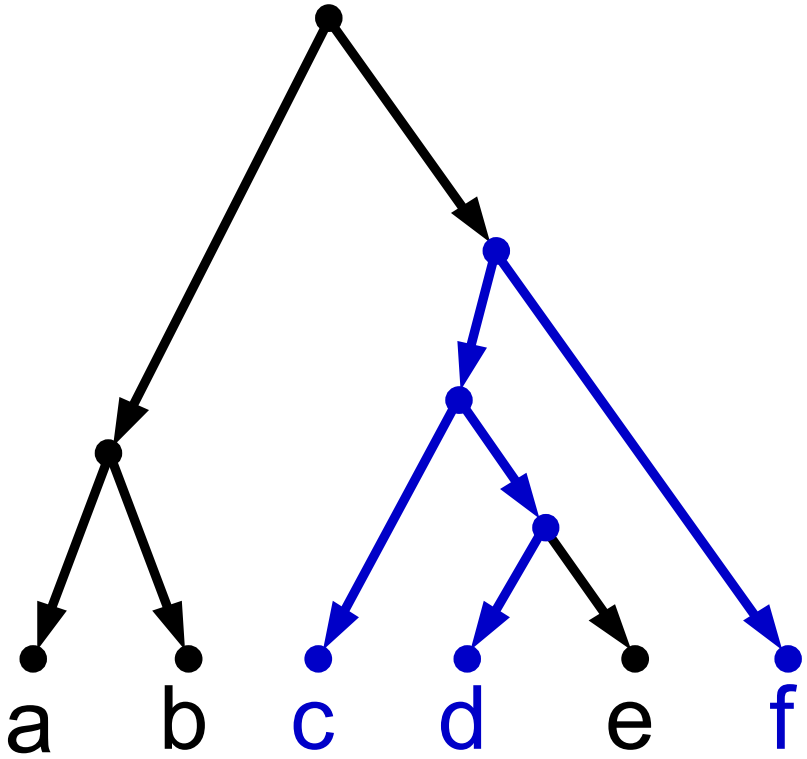


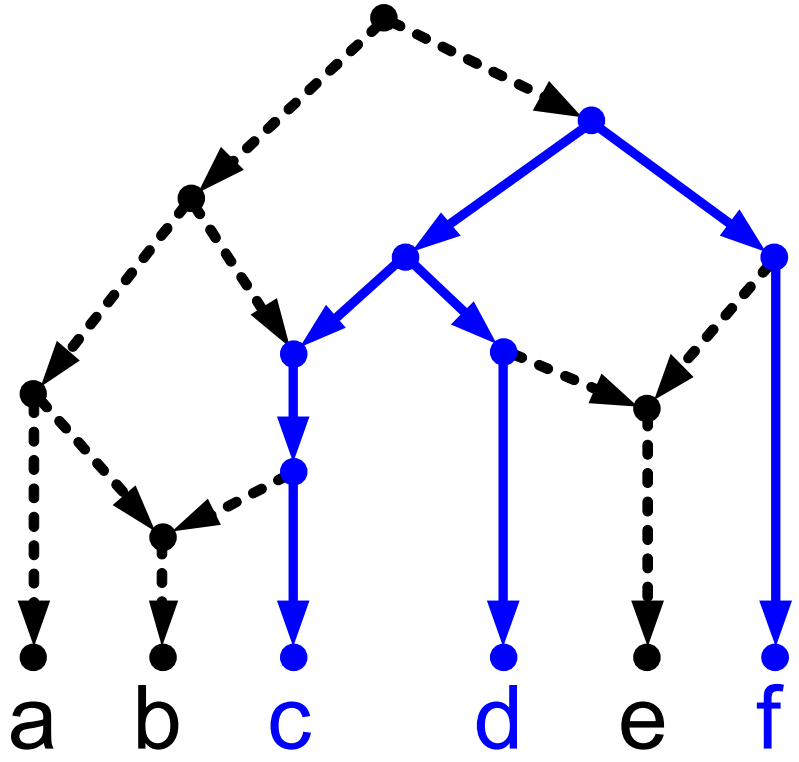
Cluster Model



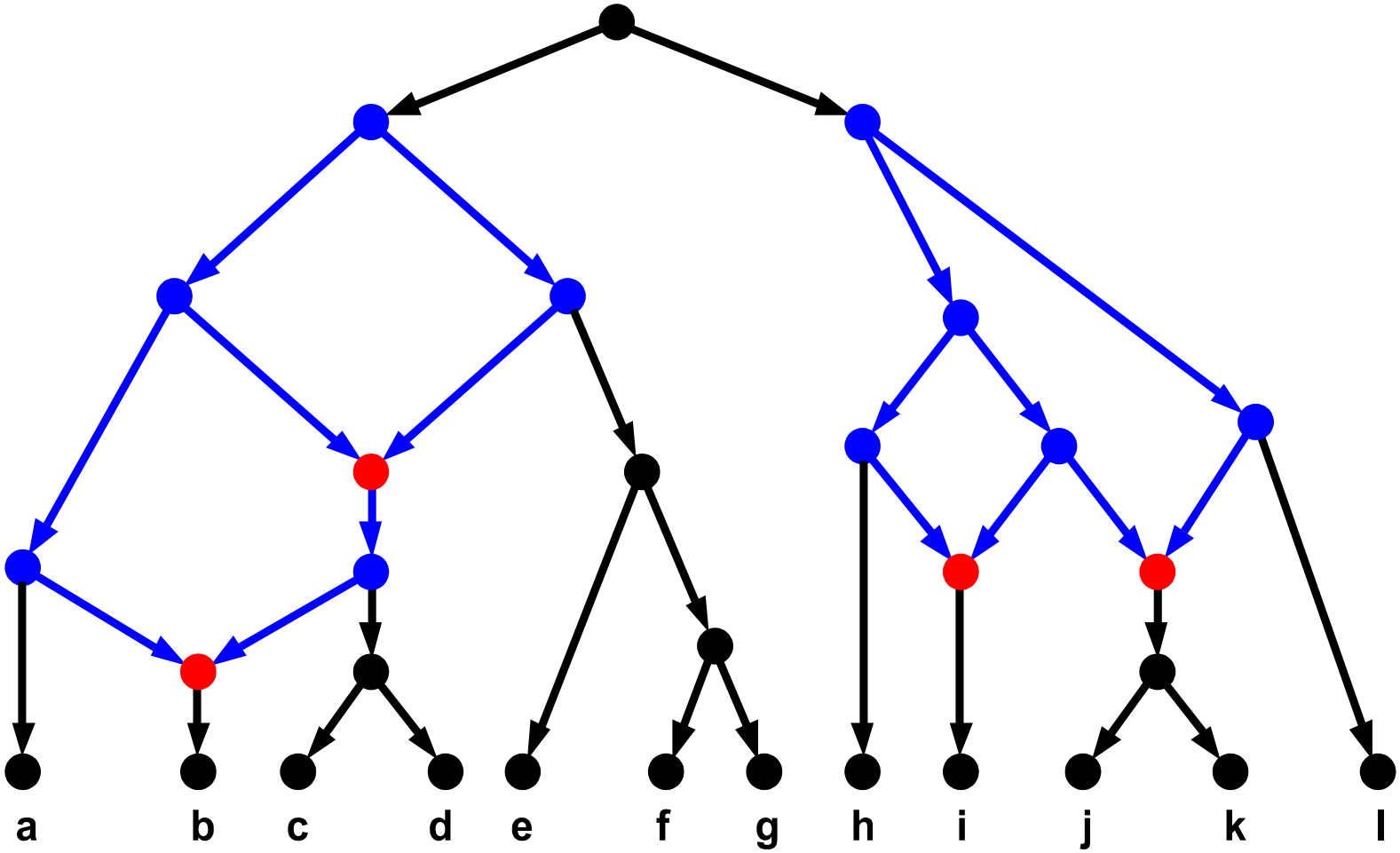


Triplet Model





Definition: the ***level*** *of a network is the maximum number of* ***reticulations*** *per* ***biconnected component***



Phylogenetic Network Problems

- **Given**: a set of **triplets/clusters/trees**
- **Find**: a **network** that represents all triplets/clusters/trees
- **Minimise**: the **level** of the network **/** total number of **reticulations**

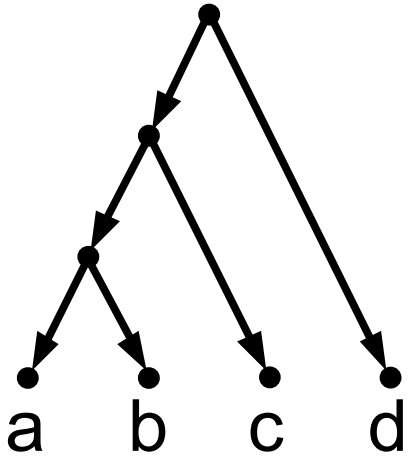
IF a network displays the trees
THEN it displays all clusters from the trees

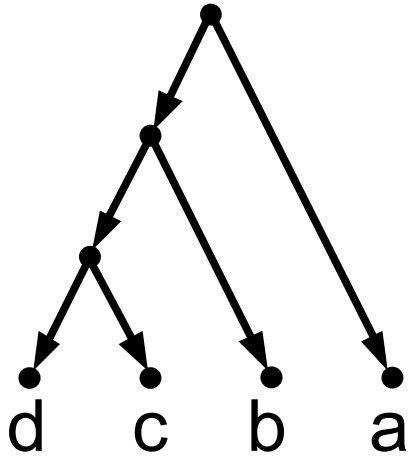
But not the other way round

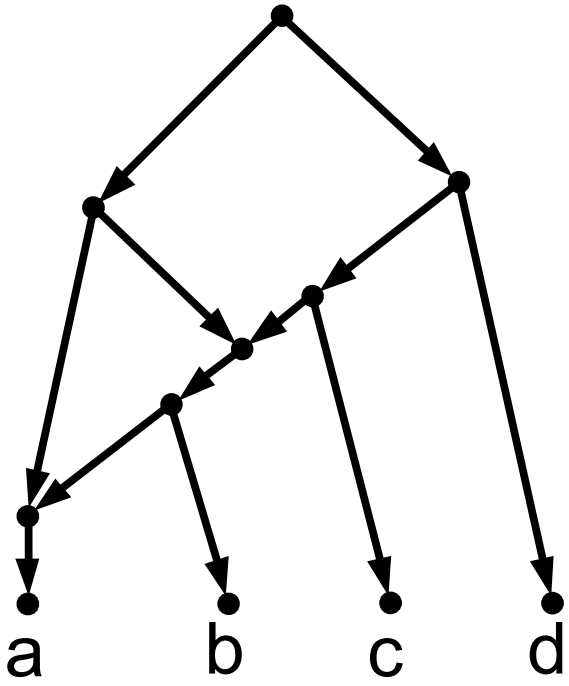
IF a network displays the trees
THEN it displays all clusters from the trees

But not the other way round

This network displays all clusters from the trees, but **not** the trees themselves





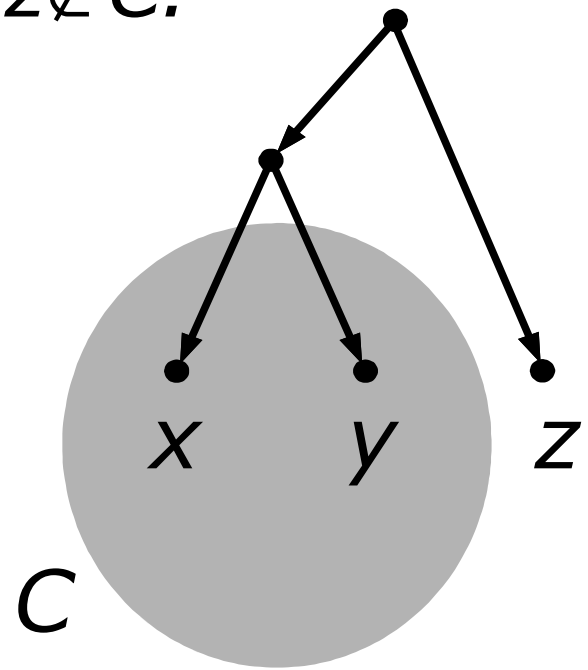


IF a network displays all clusters from the trees
THEN it displays all triplets from the trees

IF a network displays all clusters from the trees
THEN it displays all triplets from the trees

Sketch proof

If a network displays a cluster C, then it displays all triplets $xy|z$ with $x,y \in C$ and $z \notin C$.

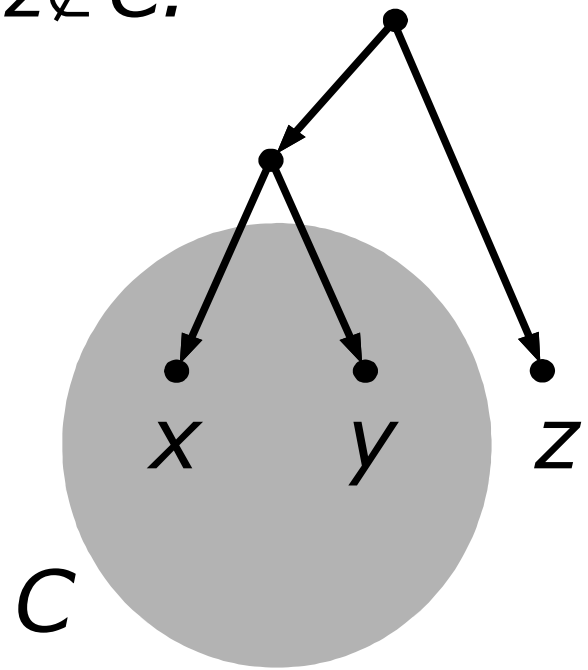


IF a network displays all clusters from the trees
THEN it displays all triplets from the trees

Sketch proof

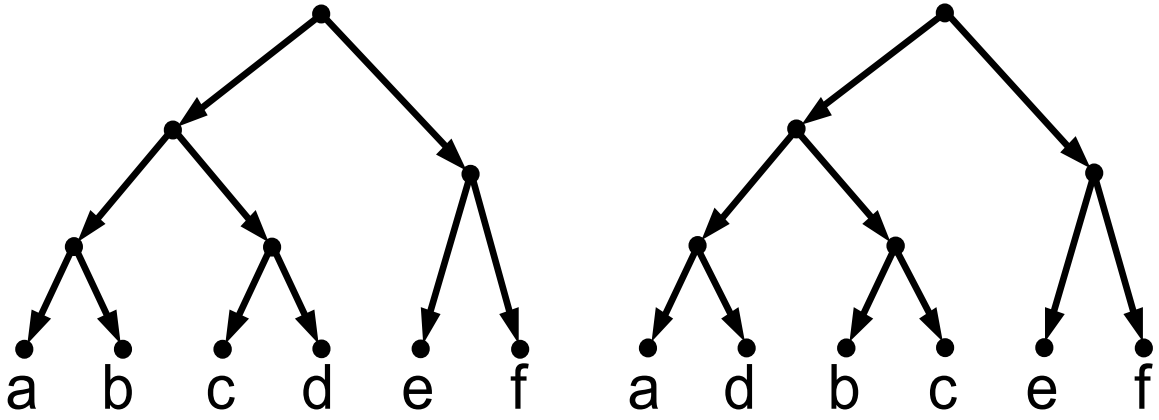
If a network displays a cluster C, then it displays all triplets $xy|z$ with $x,y \in C$ and $z \notin C$.

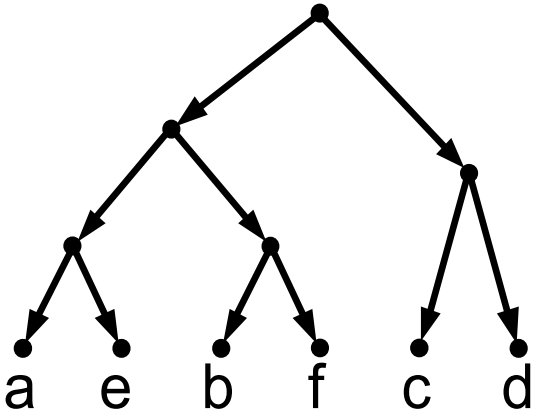
All triplets are of this form,
for some cluster C.

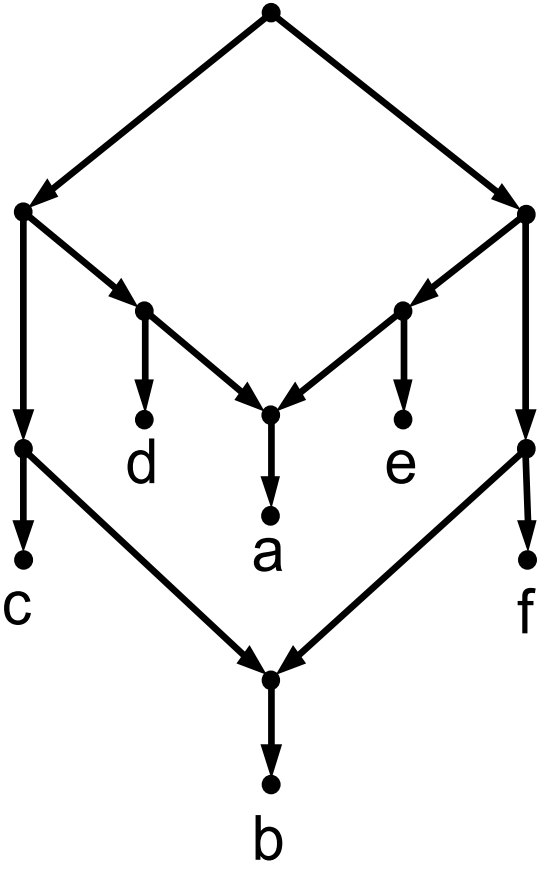


IF a network displays all clusters from the trees
THEN it displays all triplets from the trees

But not the other way round







Triplets ≤ Clusters ≤ Trees

- **In general**

Triplets ≤ Clusters ≤ Trees

- **For two trees**

Triplets = Clusters = Trees

Minimum-Level Network from Triplets

- **Given**: a set of **triplets**
- **Find**: a **network** that represents all triplets
- **Minimise**: the **level** of the network

Algorithm

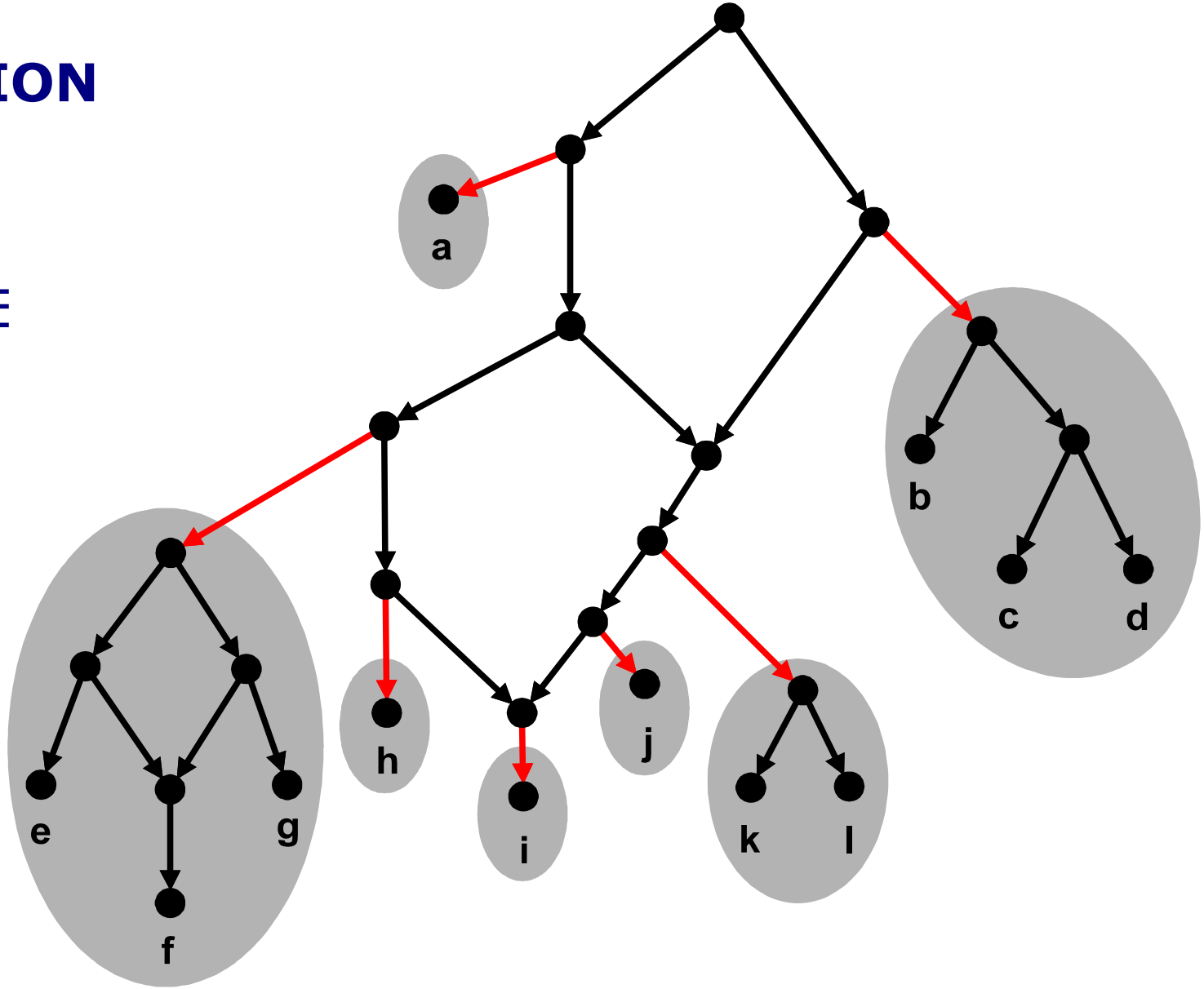
- Polynomial-time for fixed level *k*
- Input needs to contain a triplet for each combination of three leaves
- Otherwise the problem is NP-hard for every fixed *k*

Algorithm

- **PARTITION** the leaves into a correct partition
- **INDUCE** a new set of triplets where each block of the partition becomes a single superleaf
- **SOLVE** a simpler version of the problem
- **RECURSE** inside each superleaf

Algorithm

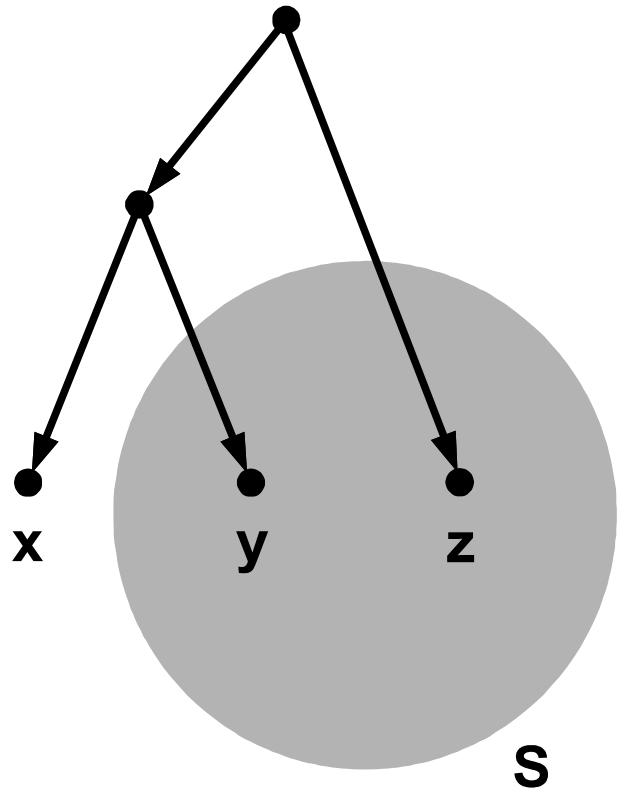
- **PARTITION**
- INDUCE
- SOLVE
- RECURSE



Algorithm

- **PARTITION**
- INDUCE
- SOLVE
- RECURSE

- **Definition.** Subset S of the leaves is an ***SN-set*** if there is no triplet xy|z with y,z∈ S and x∉ S



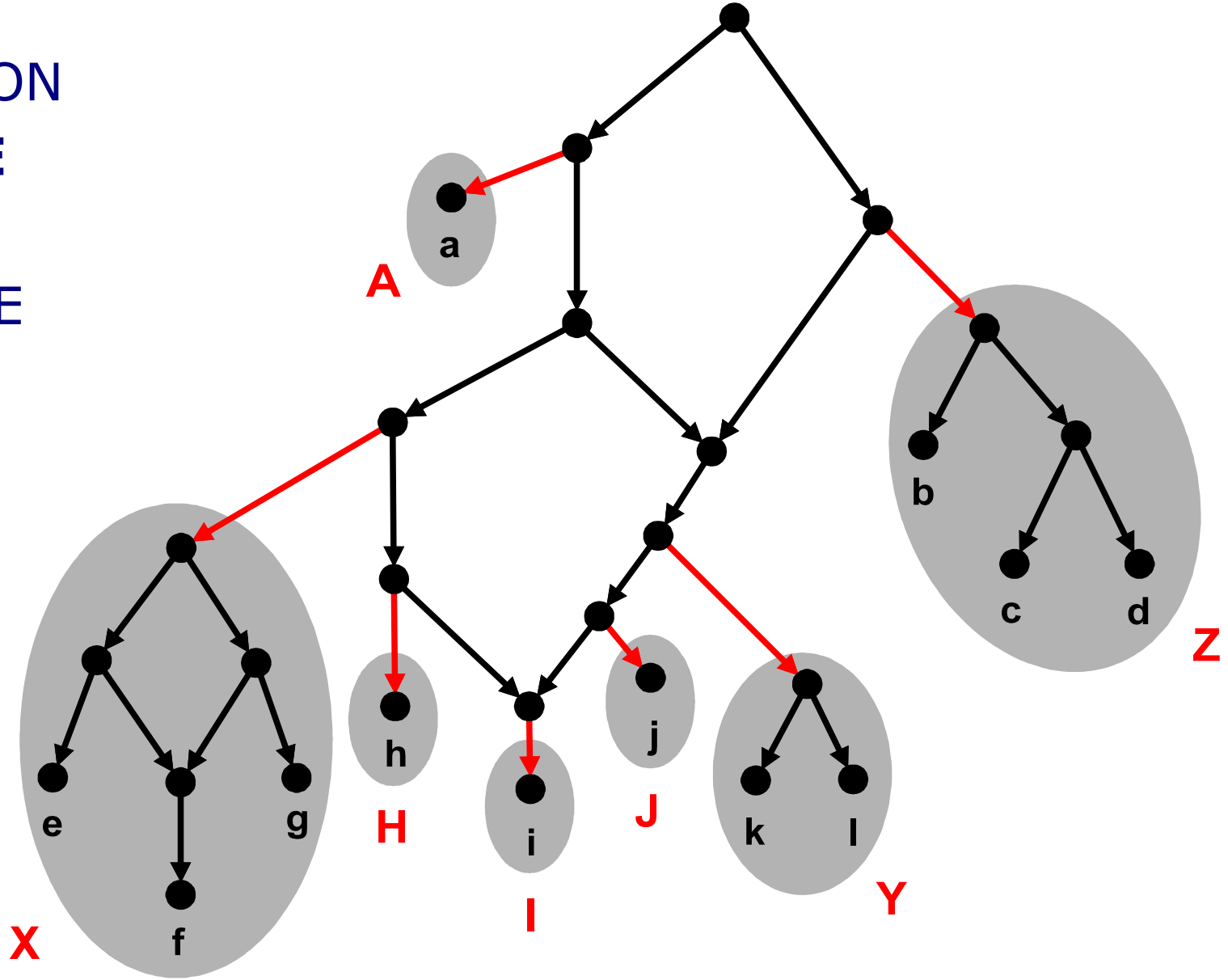
Algorithm

- **PARTITION**
- INDUCE
- SOLVE
- RECURSE

- For **level-1**: maximal SN-sets form a correct partition
- For **level-2**: at most **one** SN-set needs to be split
- For **level-*k***: at most ***f(k)*** SN-sets need to be split

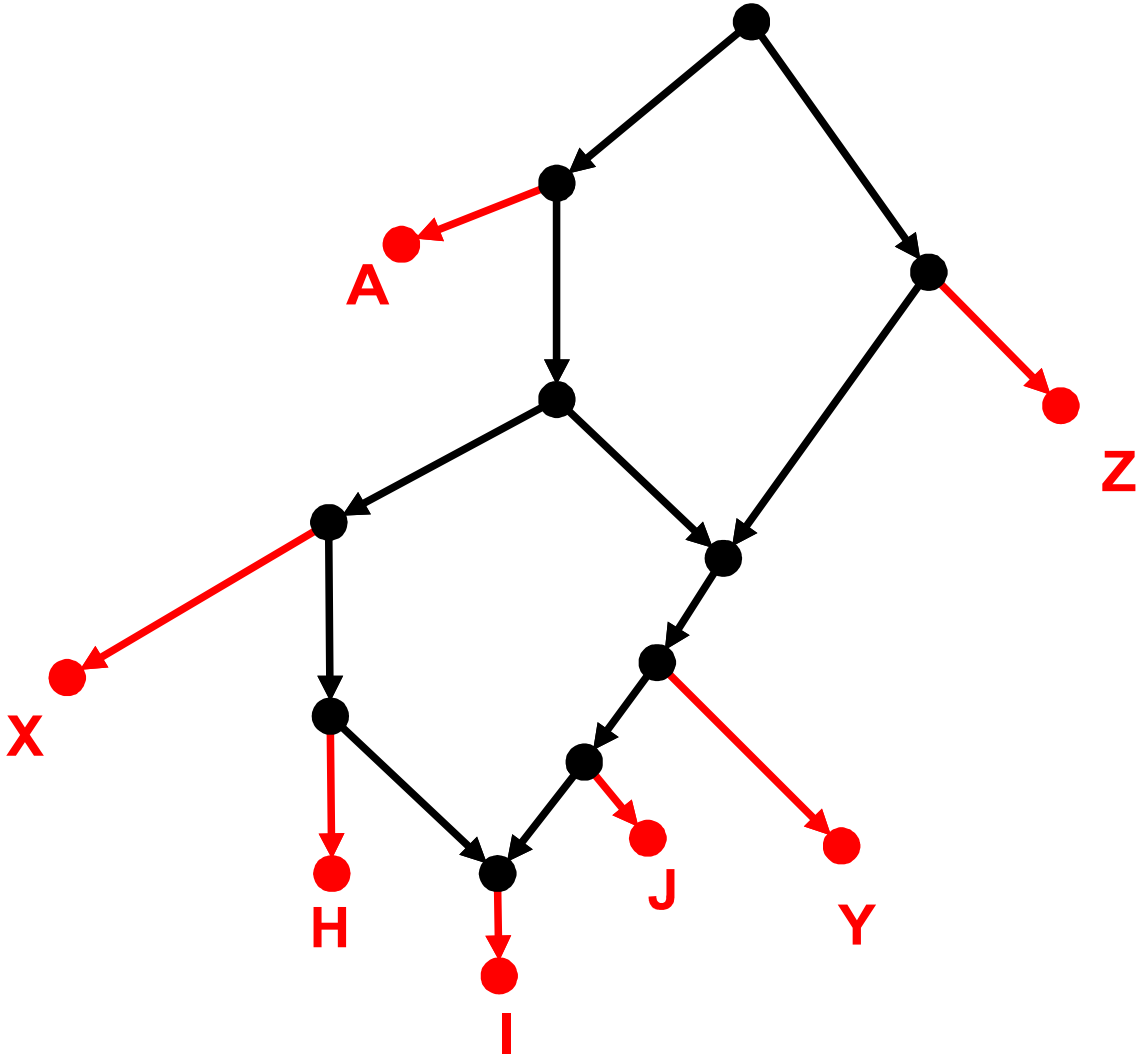
Algorithm

- PARTITION
- **INDUCE**
- SOLVE
- RECURSE



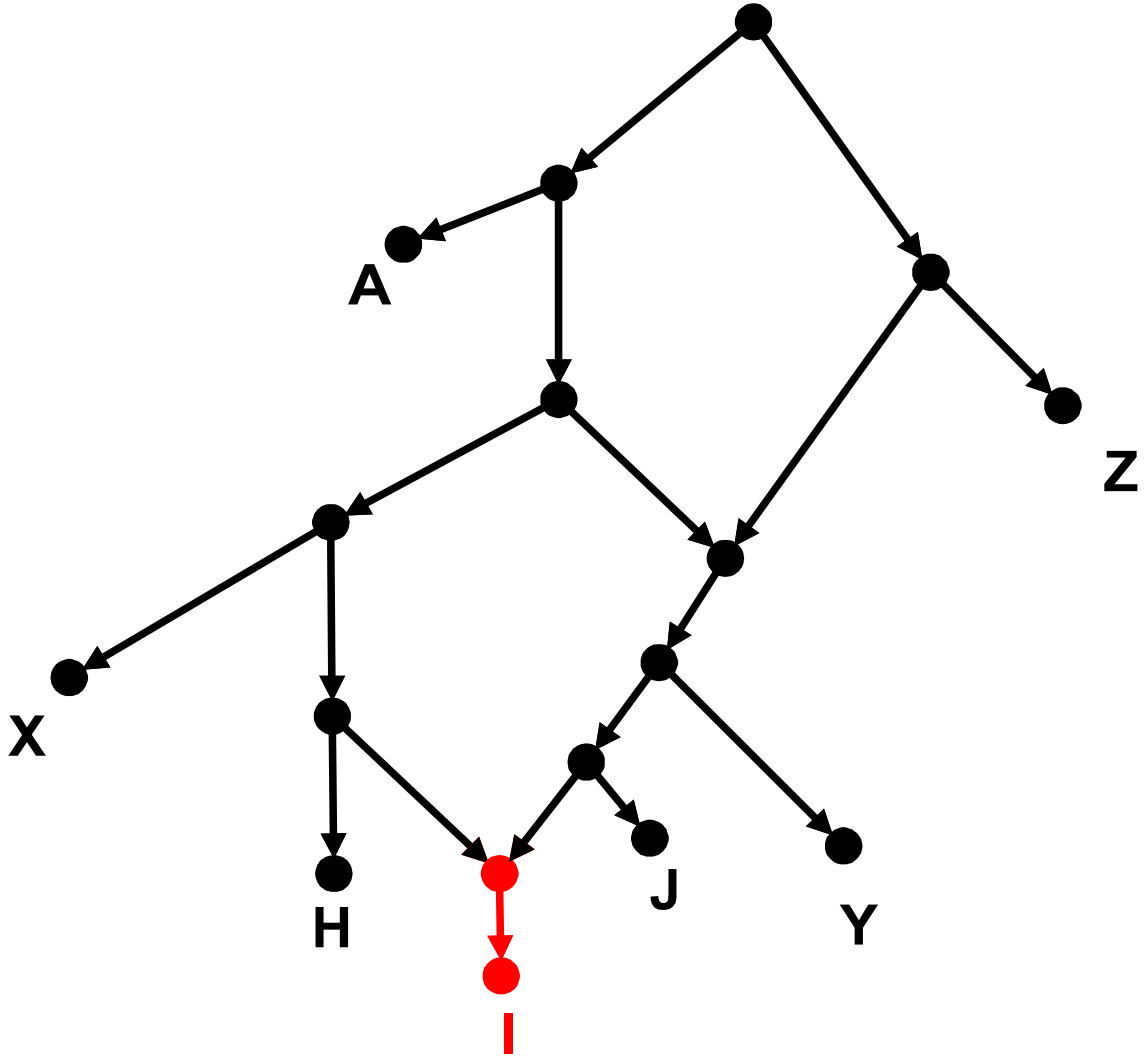
Algorithm

- PARTITION
- **INDUCE**
- SOLVE
- RECURSE



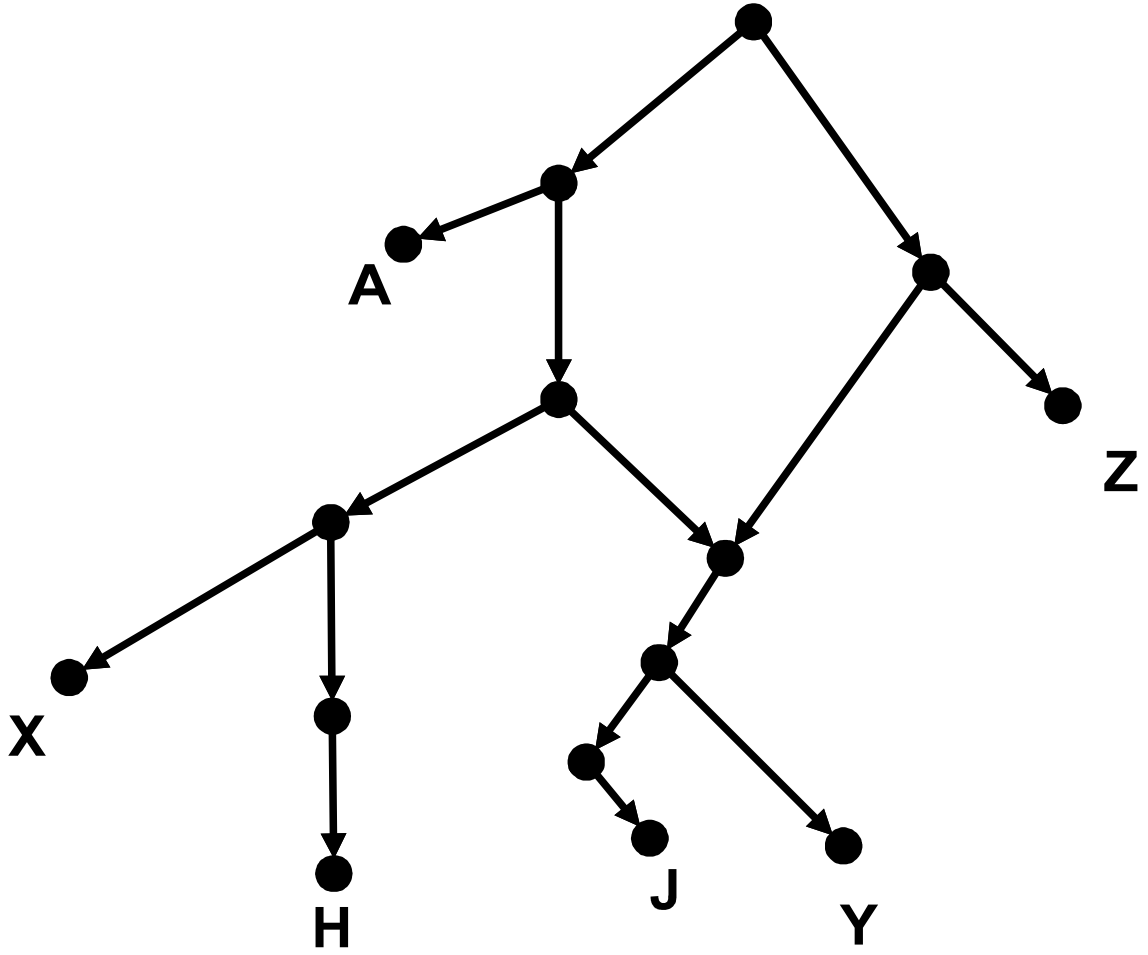
Algorithm

- PARTITION
- INDUCE
- **SOLVE**
- RECURSE



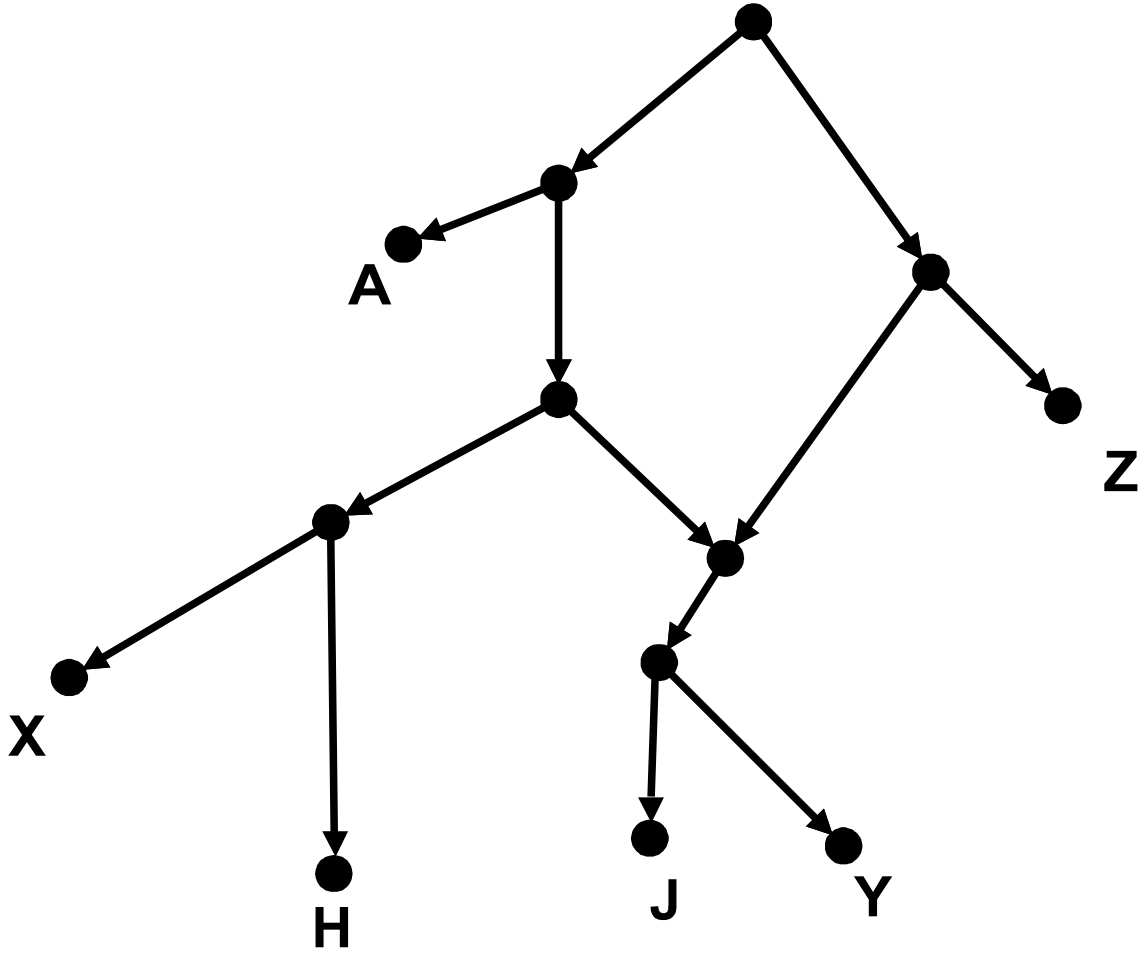
Algorithm

- PARTITION
- INDUCE
- **SOLVE**
- RECURSE



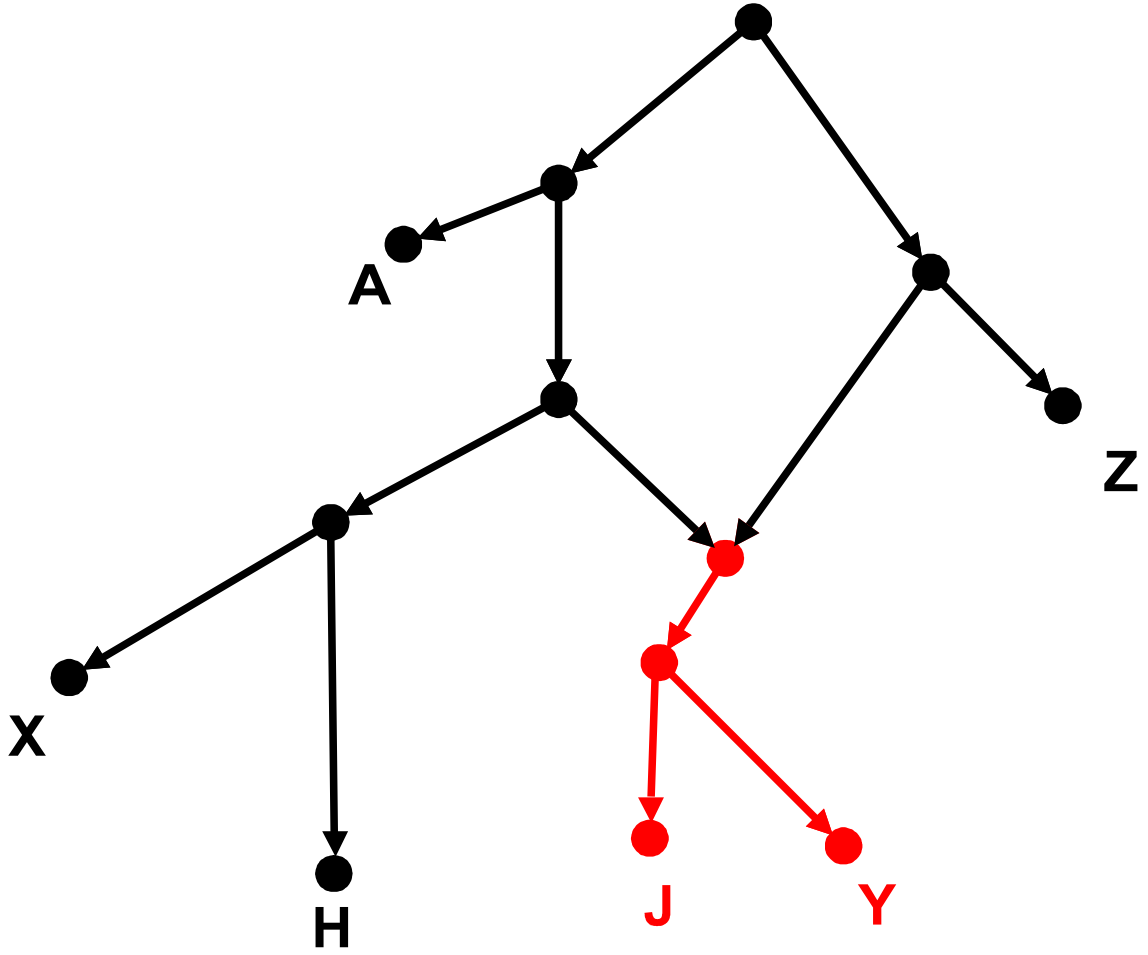
Algorithm

- PARTITION
- INDUCE
- **SOLVE**
- RECURSE



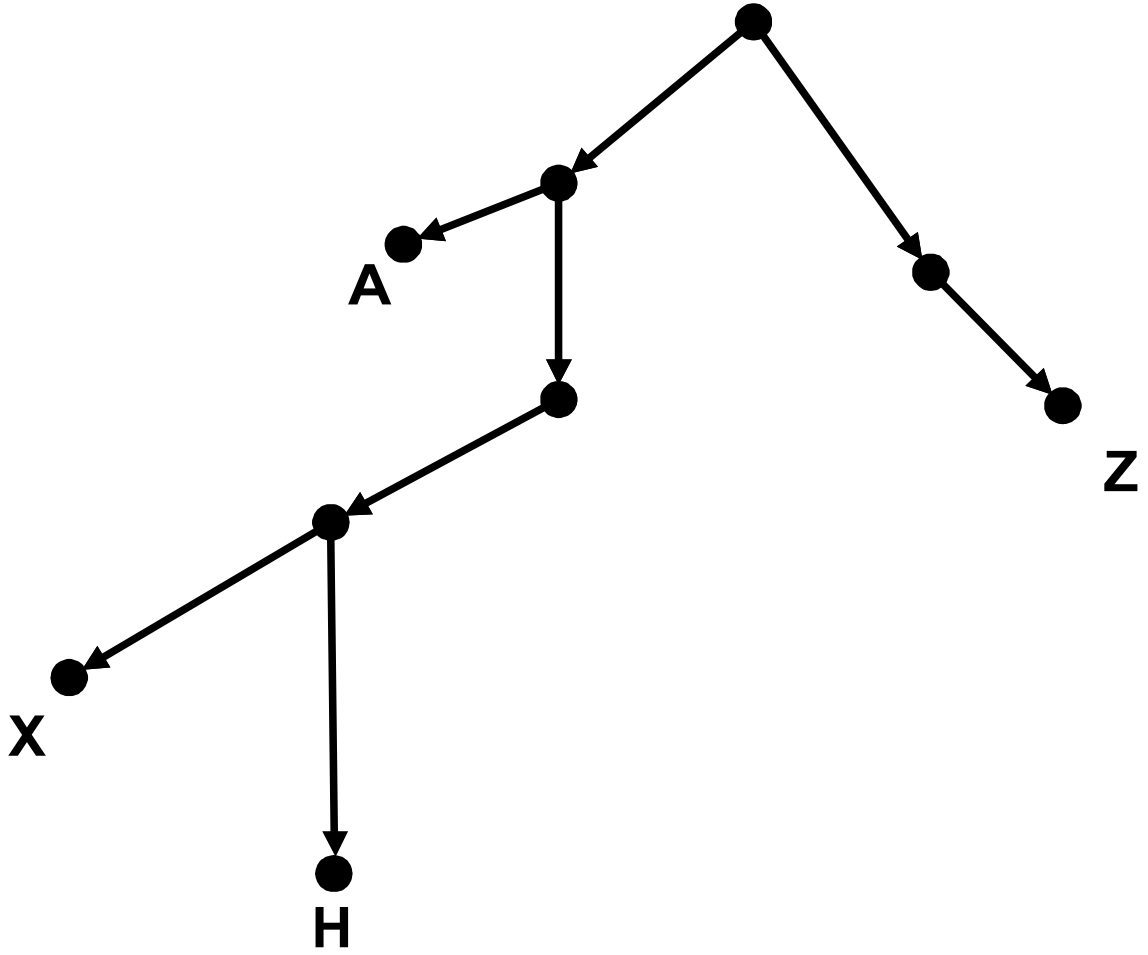
Algorithm

- PARTITION
- INDUCE
- **SOLVE**
- RECURSE



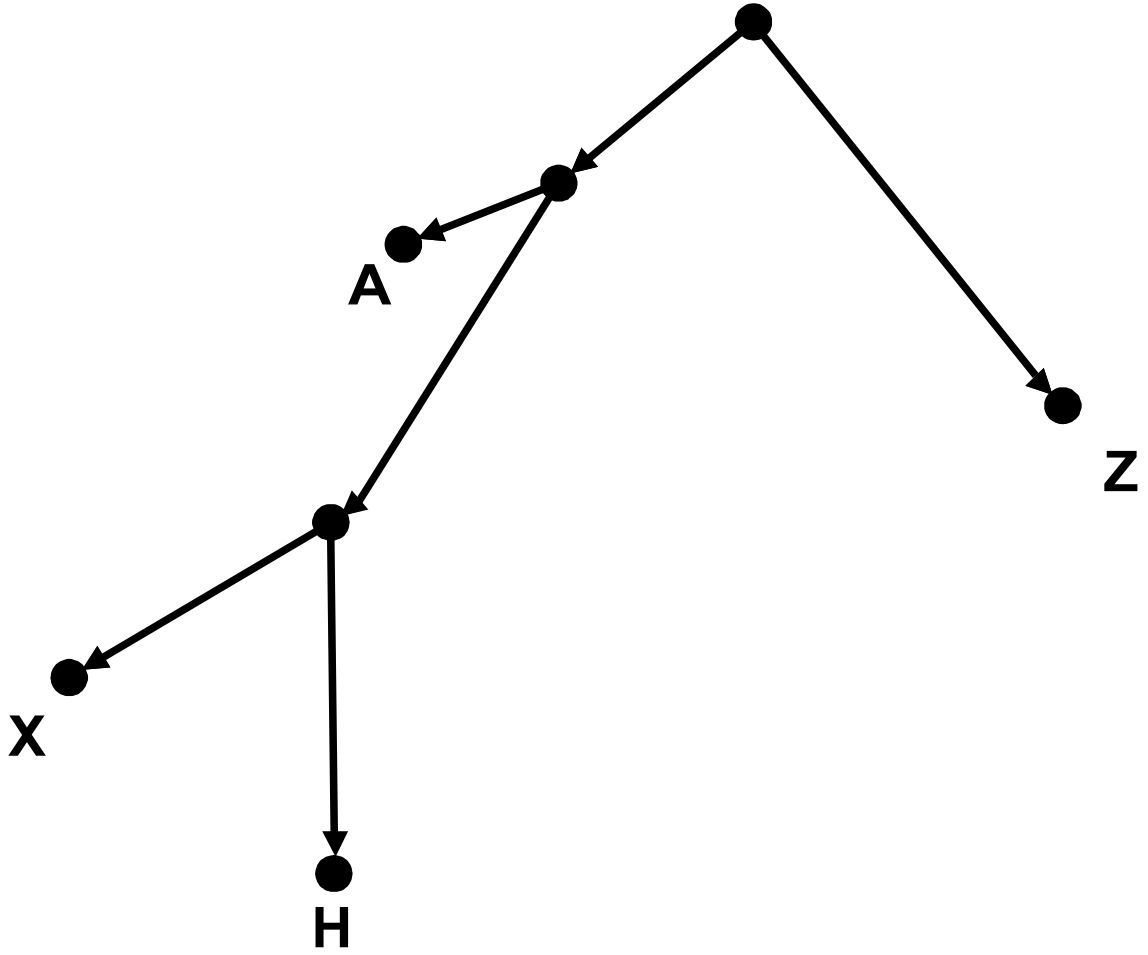
Algorithm

- PARTITION
- INDUCE
- **SOLVE**
- RECURSE



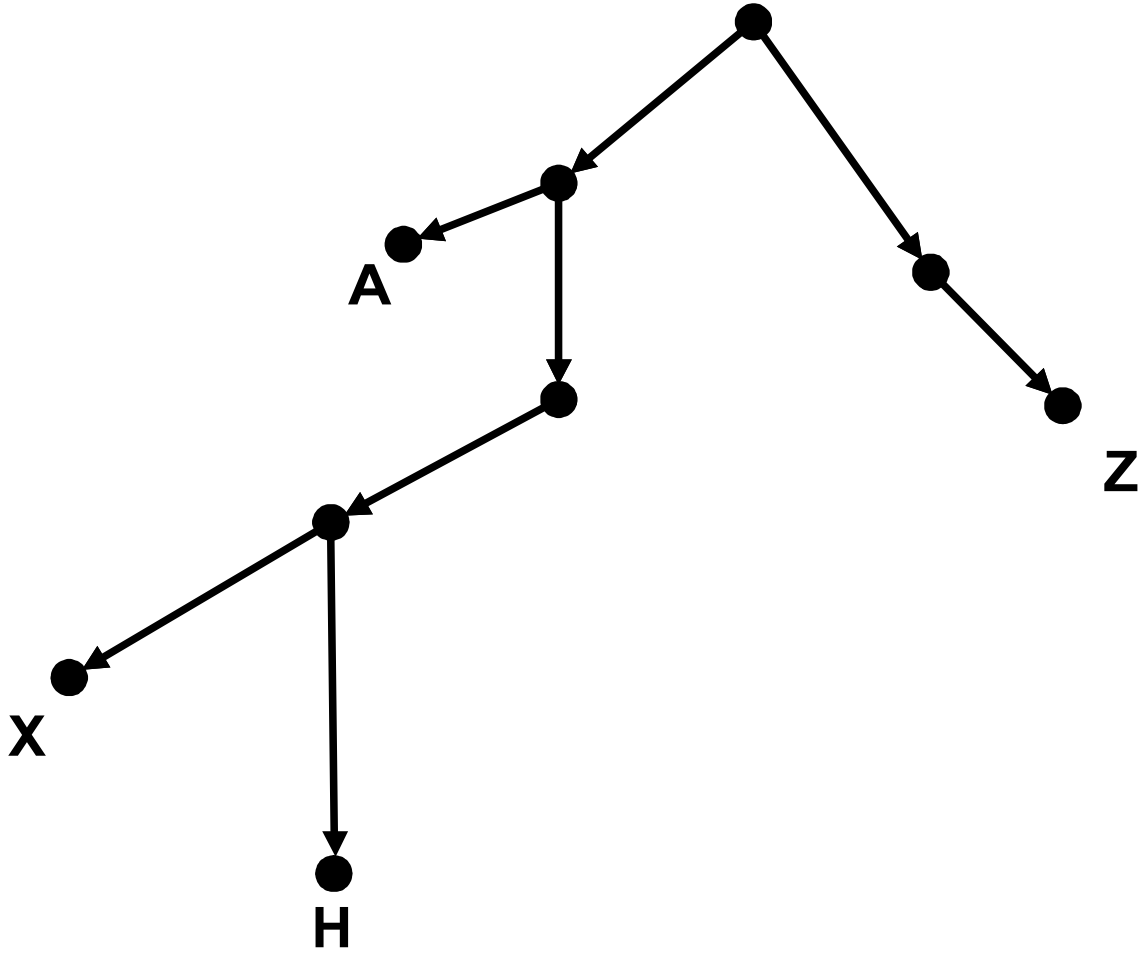
Algorithm

- PARTITION
- INDUCE
- **SOLVE**
- RECURSE



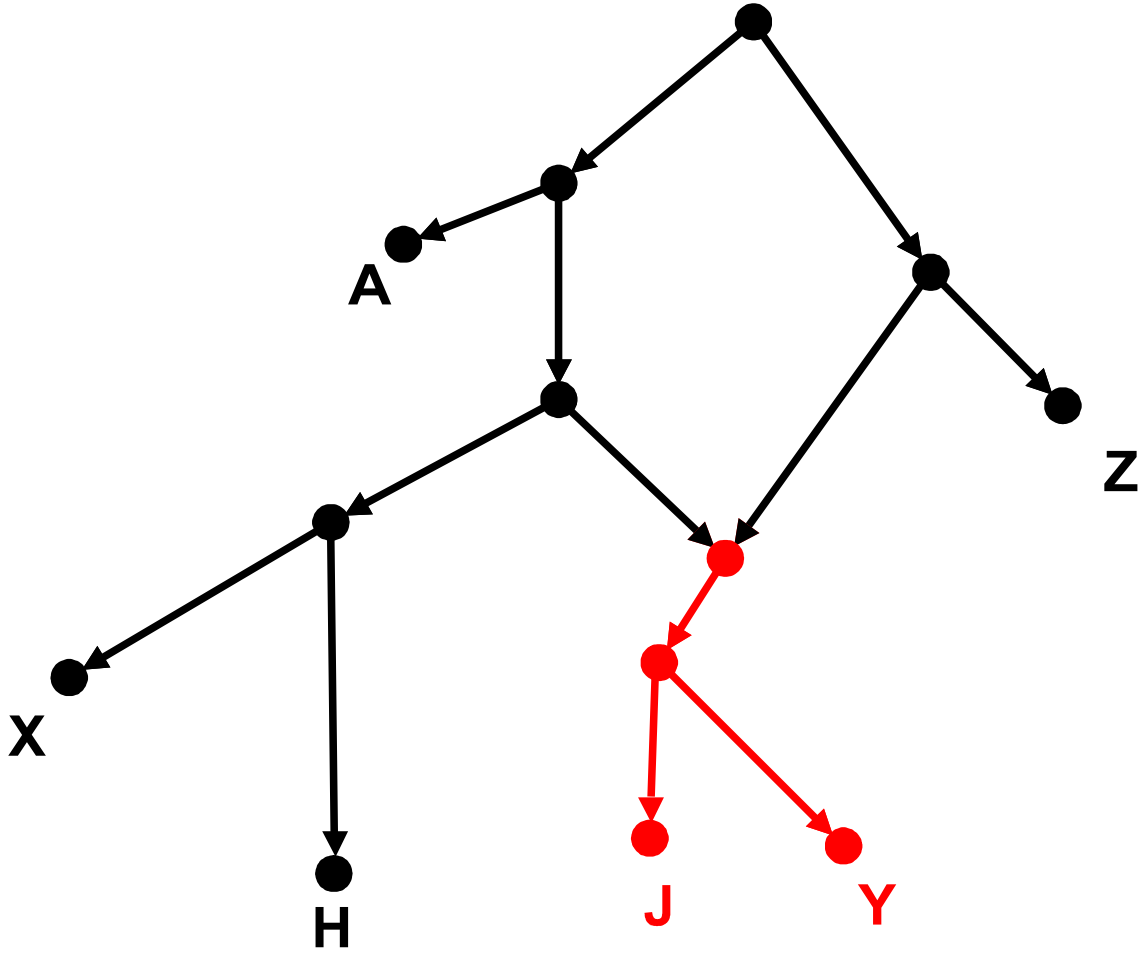
Algorithm

- PARTITION
- INDUCE
- **SOLVE**
- RECURSE



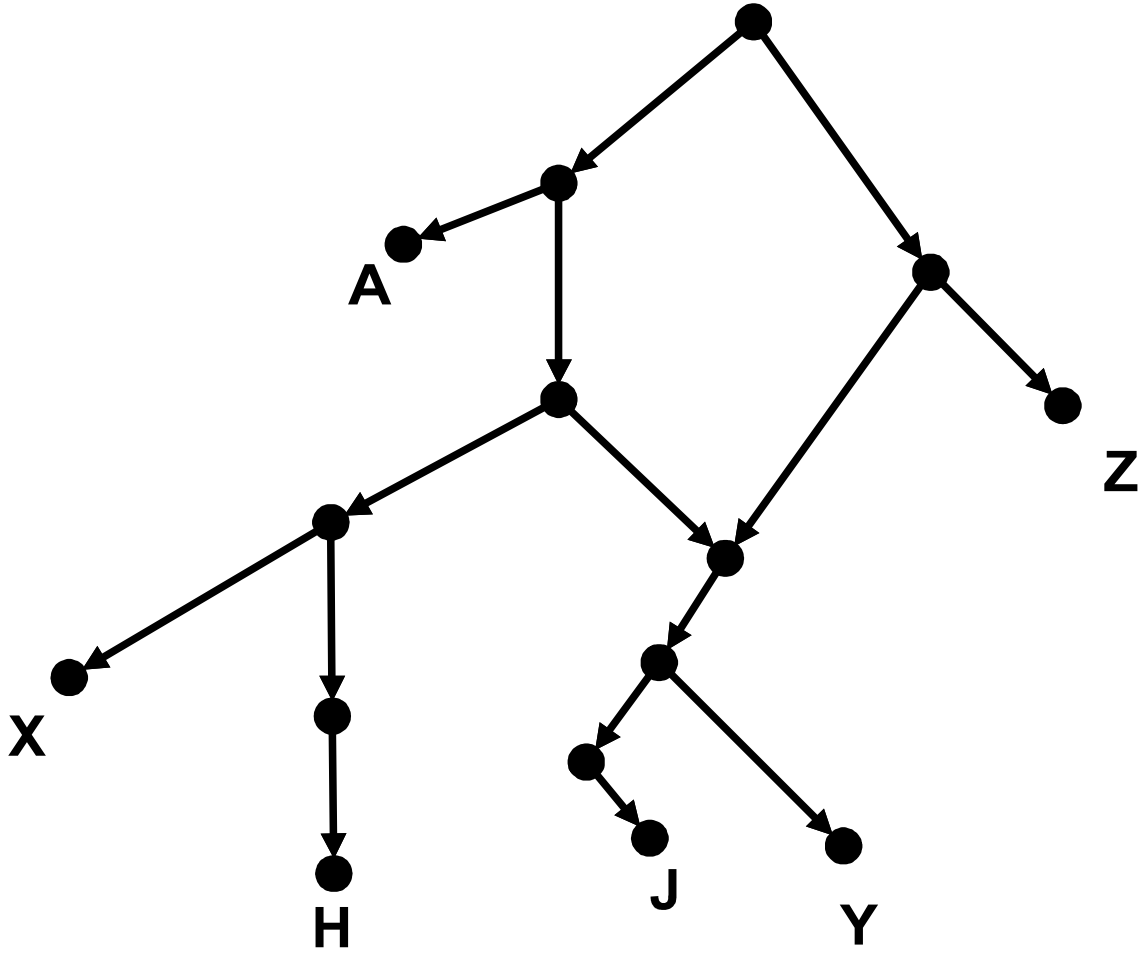
Algorithm

- PARTITION
- INDUCE
- **SOLVE**
- RECURSE



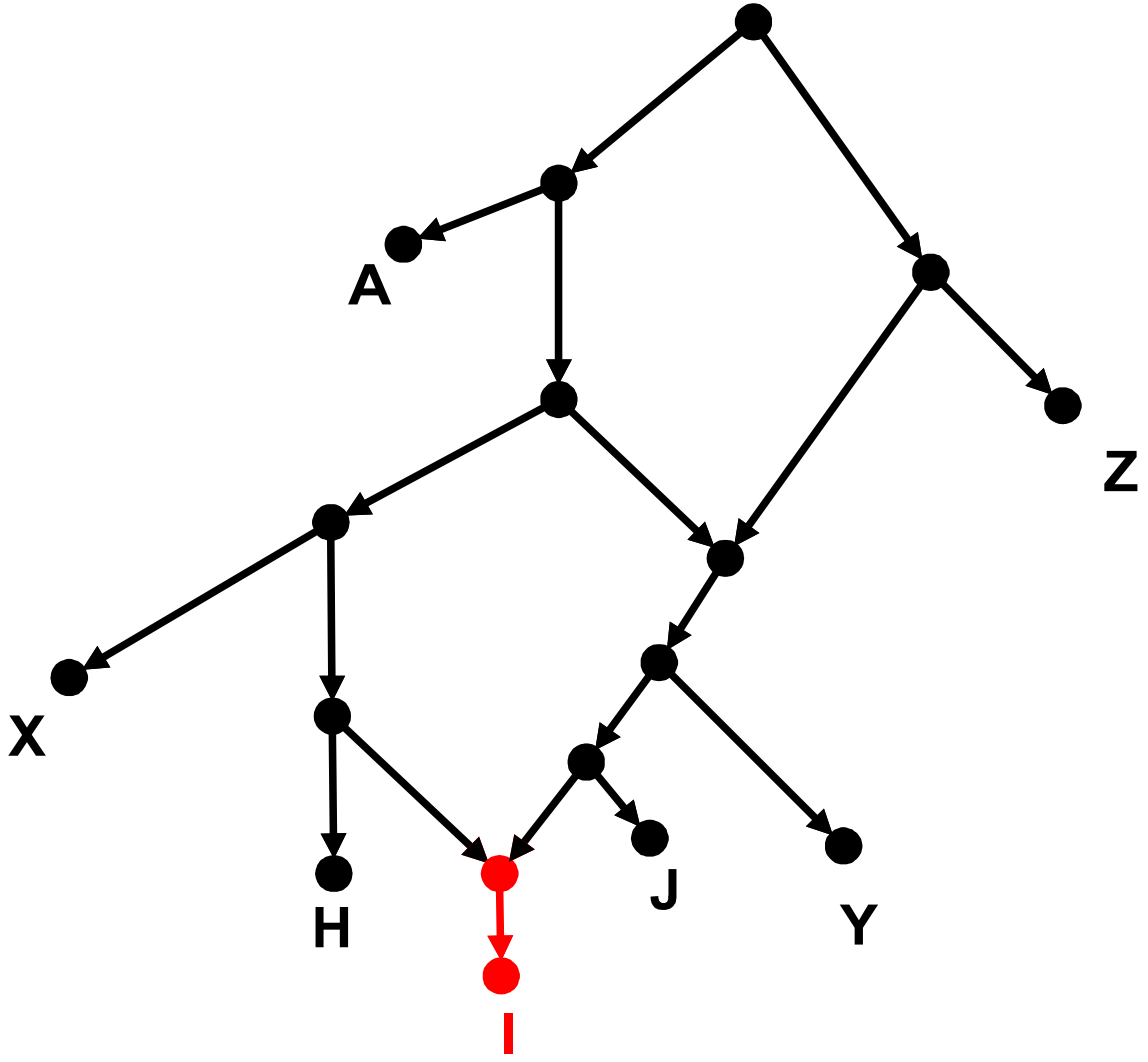
Algorithm

- PARTITION
- INDUCE
- **SOLVE**
- RECURSE



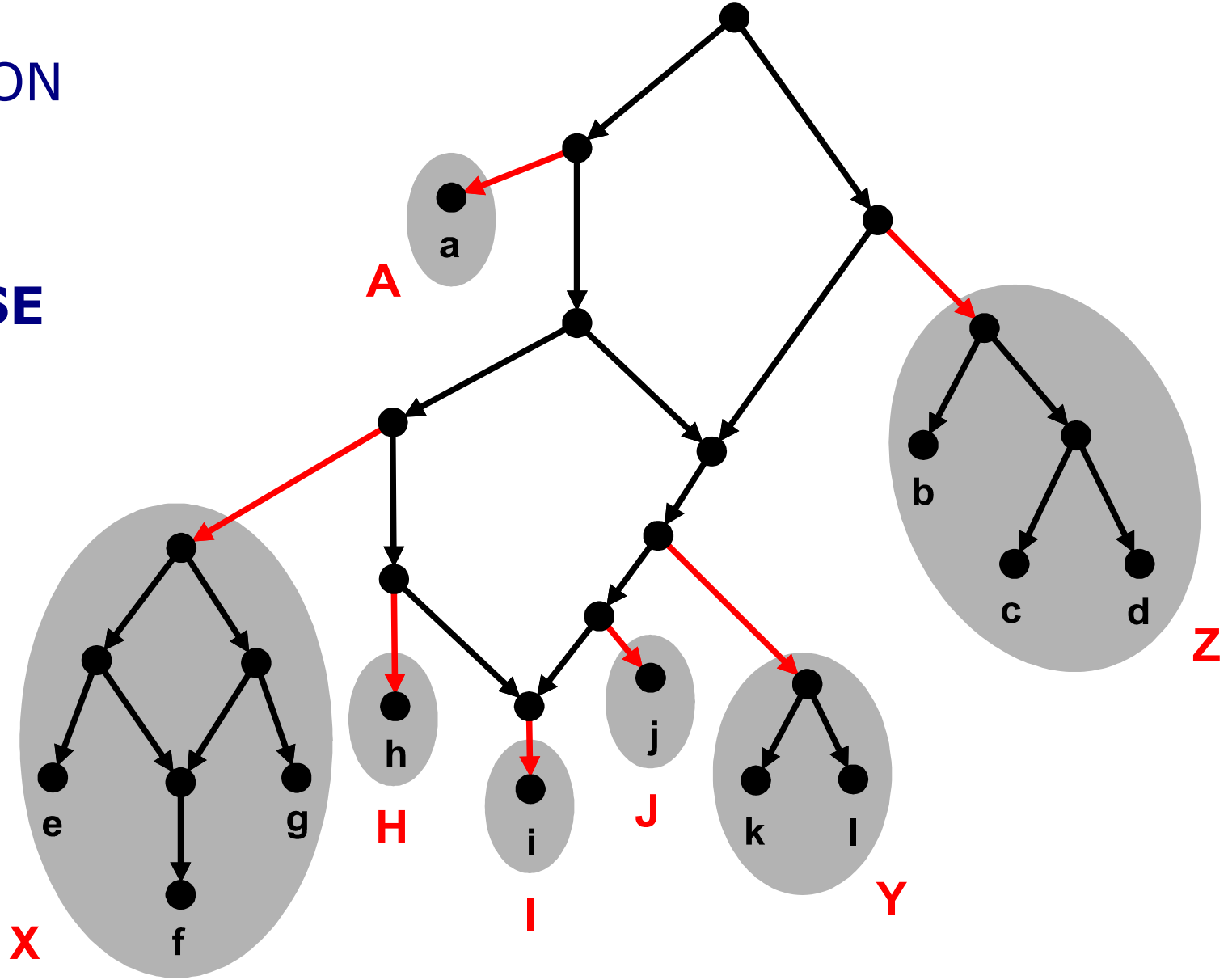
Algorithm

- PARTITION
- INDUCE
- **SOLVE**
- RECURSE



Algorithm

- PARTITION
- INDUCE
- SOLVE
- **RECURSE**



Algorithms for Combining Phylogenetic Trees into a Network