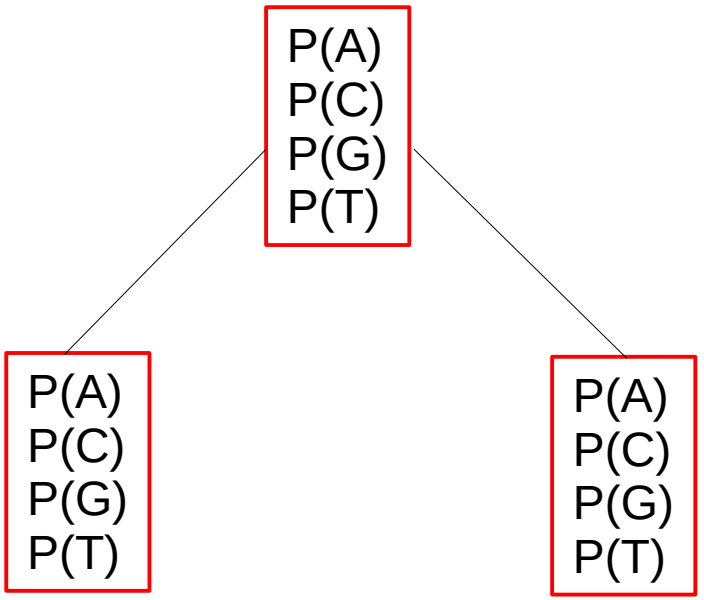
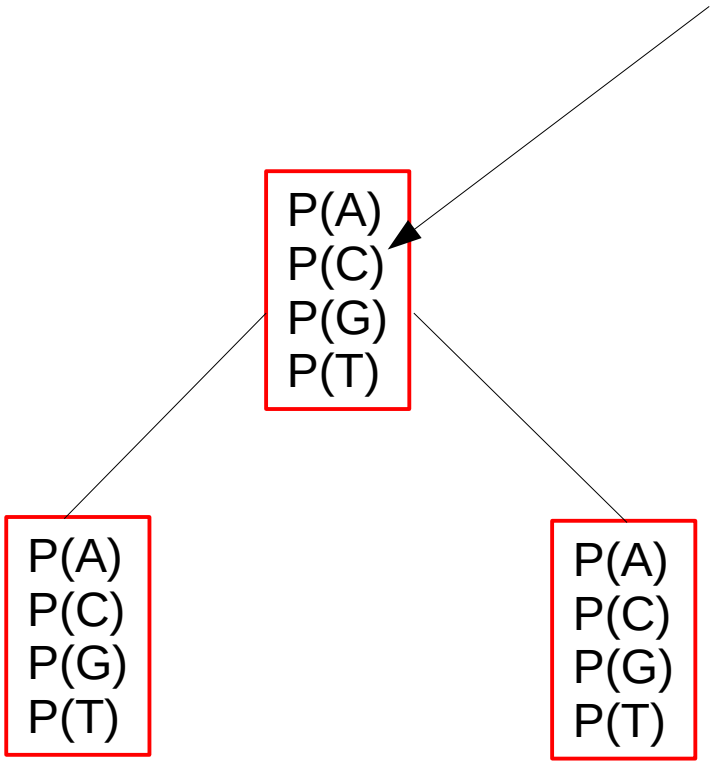
Introduction to Bioinformatics for Computer Scientists

Lecture 10

Scaling

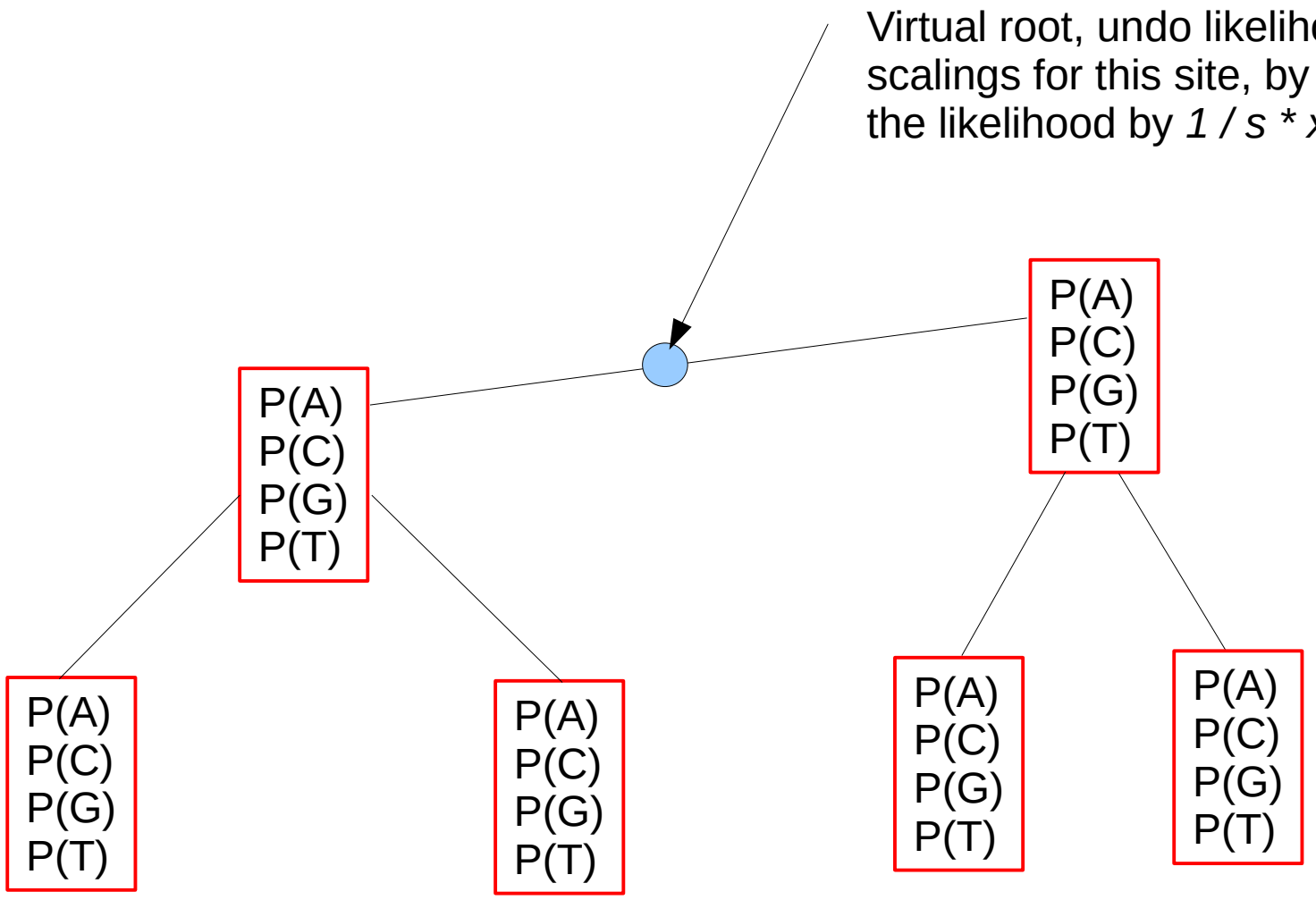


Scaling

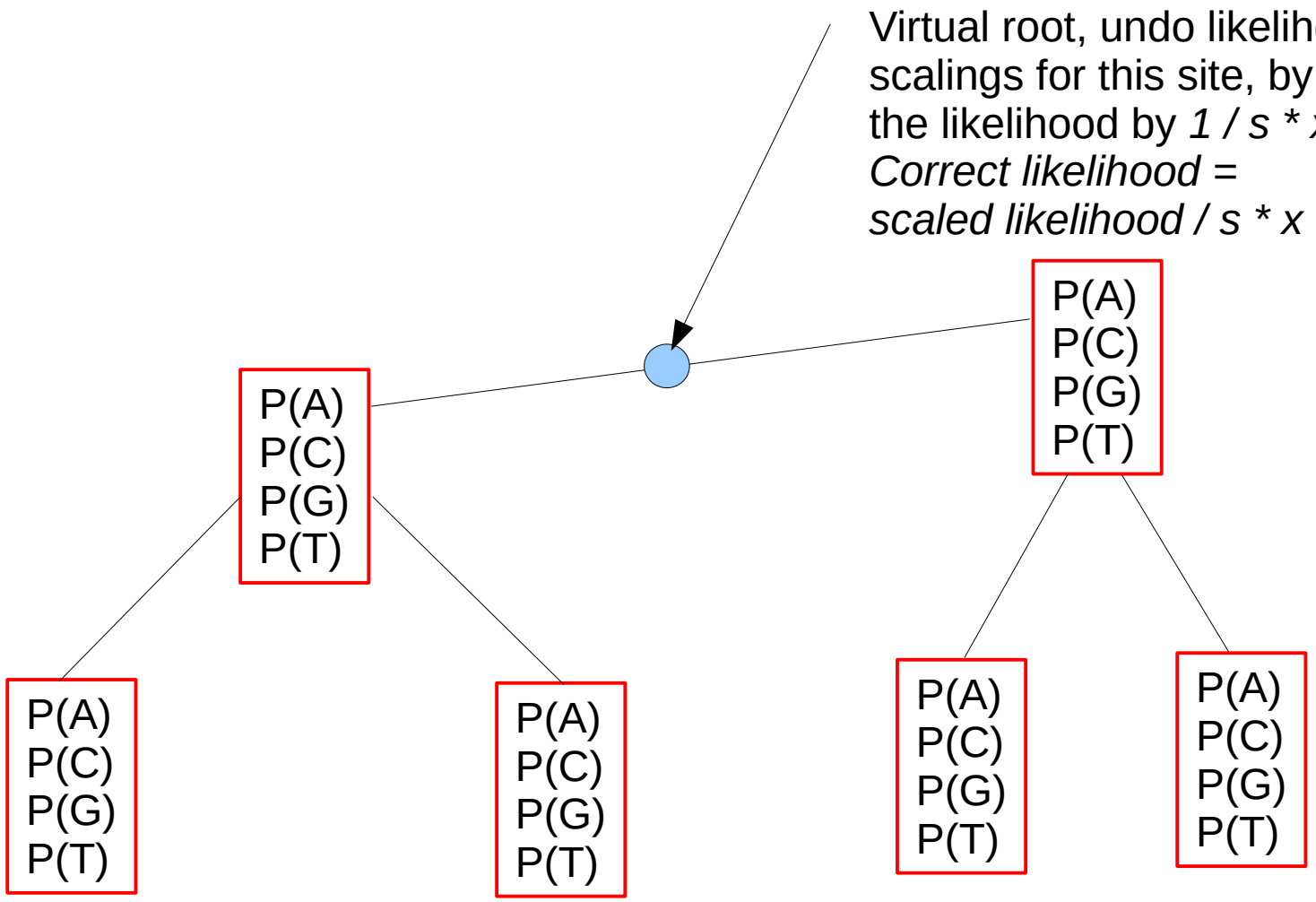


Check if one or more entries
Are too small.
If yes, multiply with some large
Constant number *x* and increment
a per-site scaling counter *s, s = s + 1*

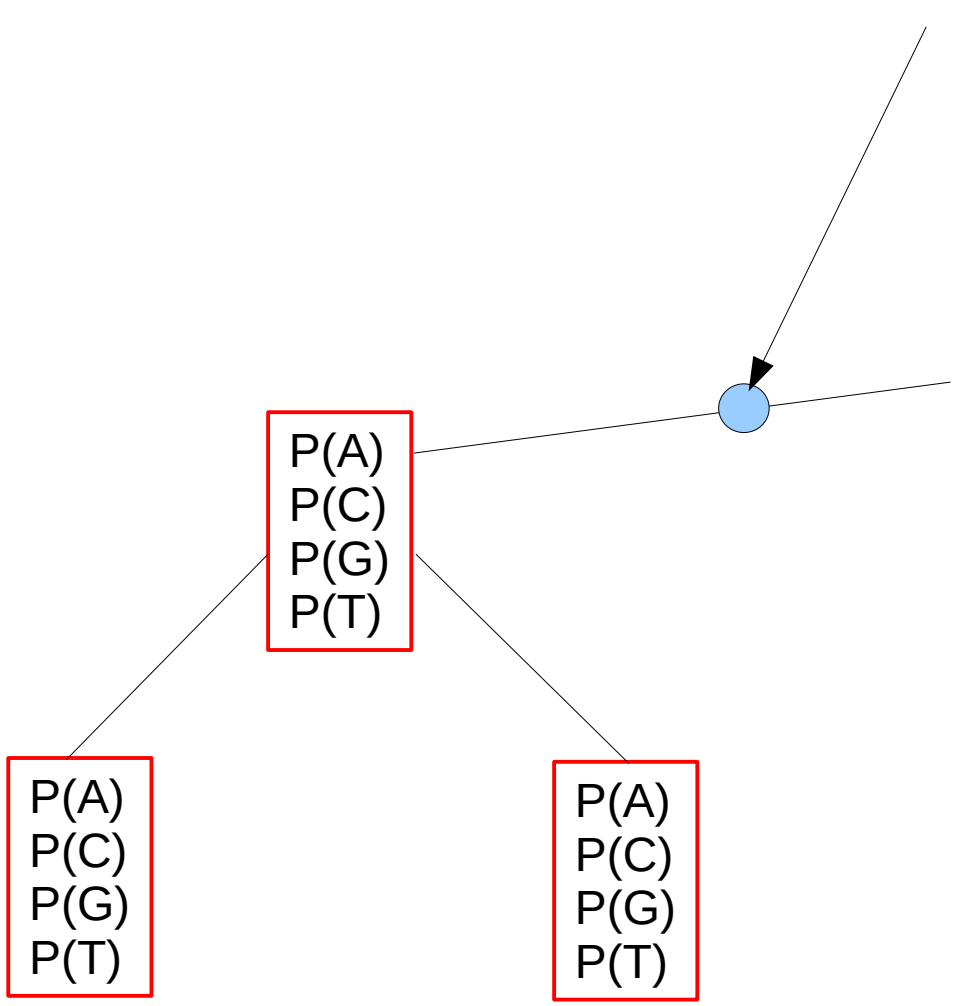
Scaling


Virtual root, undo likelihood scalings for this site, by dividing the likelihood by *1 / s * x*
P(A)
P(C)
P(G)
P(T)
P(A)
P(C)
P(G)
P(T)
P(A)
P(C)
P(G)
P(T)
P(A)
P(C)
P(G)
P(T)
P(A)
P(C)
P(G)
P(T)

Scaling


Virtual root, undo likelihood scalings for this site, by dividing the likelihood by *1 / s * x* → *Correct likelihood = scaled likelihood / s * x*
P(A)
P(C)
P(G)
P(T)
P(A)
P(C)
P(G)
P(T)
P(A)
P(C)
P(G)
P(T)
P(A)
P(C)
P(G)
P(T)
P(A)
P(C)
P(G)
P(T)

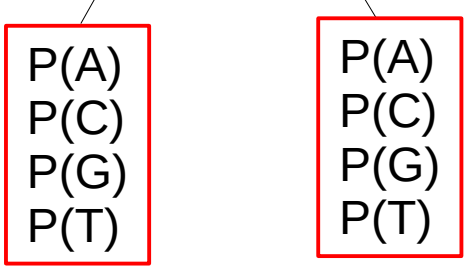
Scaling



Virtual root, undo likelihood scalings for this site, by dividing the likelihood by *1 / s * x* →
Correct likelihood =
*scaled likelihood / s * x*

Taking the logarithm:

*log(cl) = log(sl / s * x) =*
*log(sl) – log(s * x) =*
log(sl) – (log(s) + log(x))

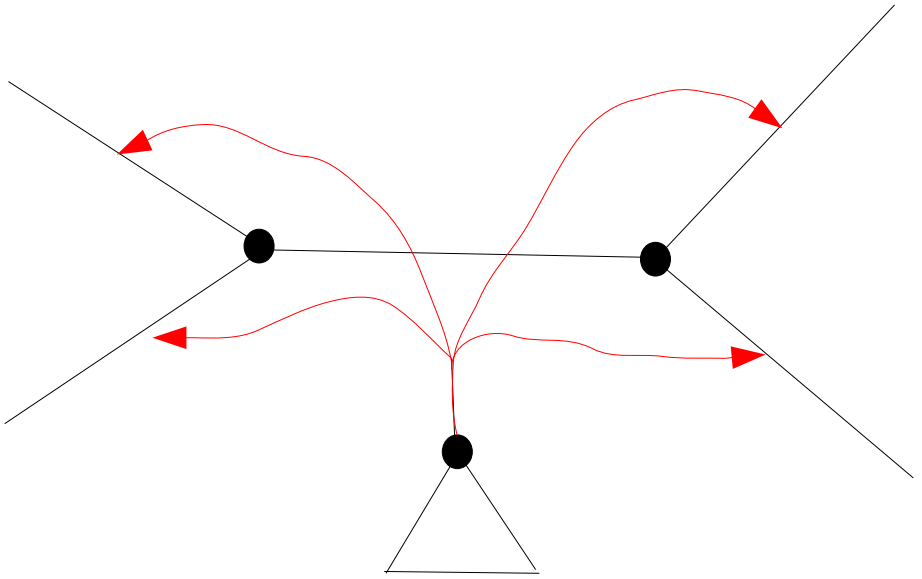


Why not parallelize by subtrees?

- Short answer: sequential dependencies & memory organization

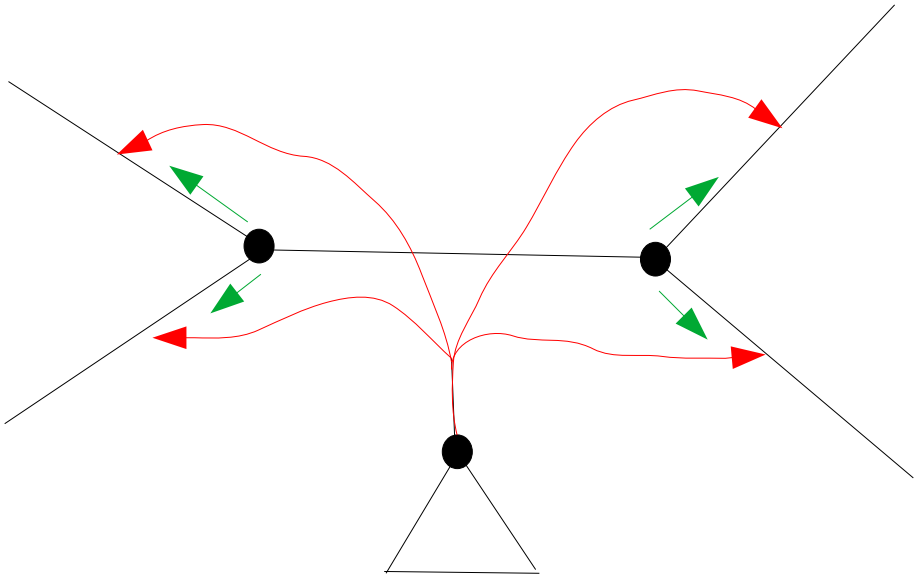
Why not parallelize by subtrees?

- Short answer: sequential dependencies & memory organization



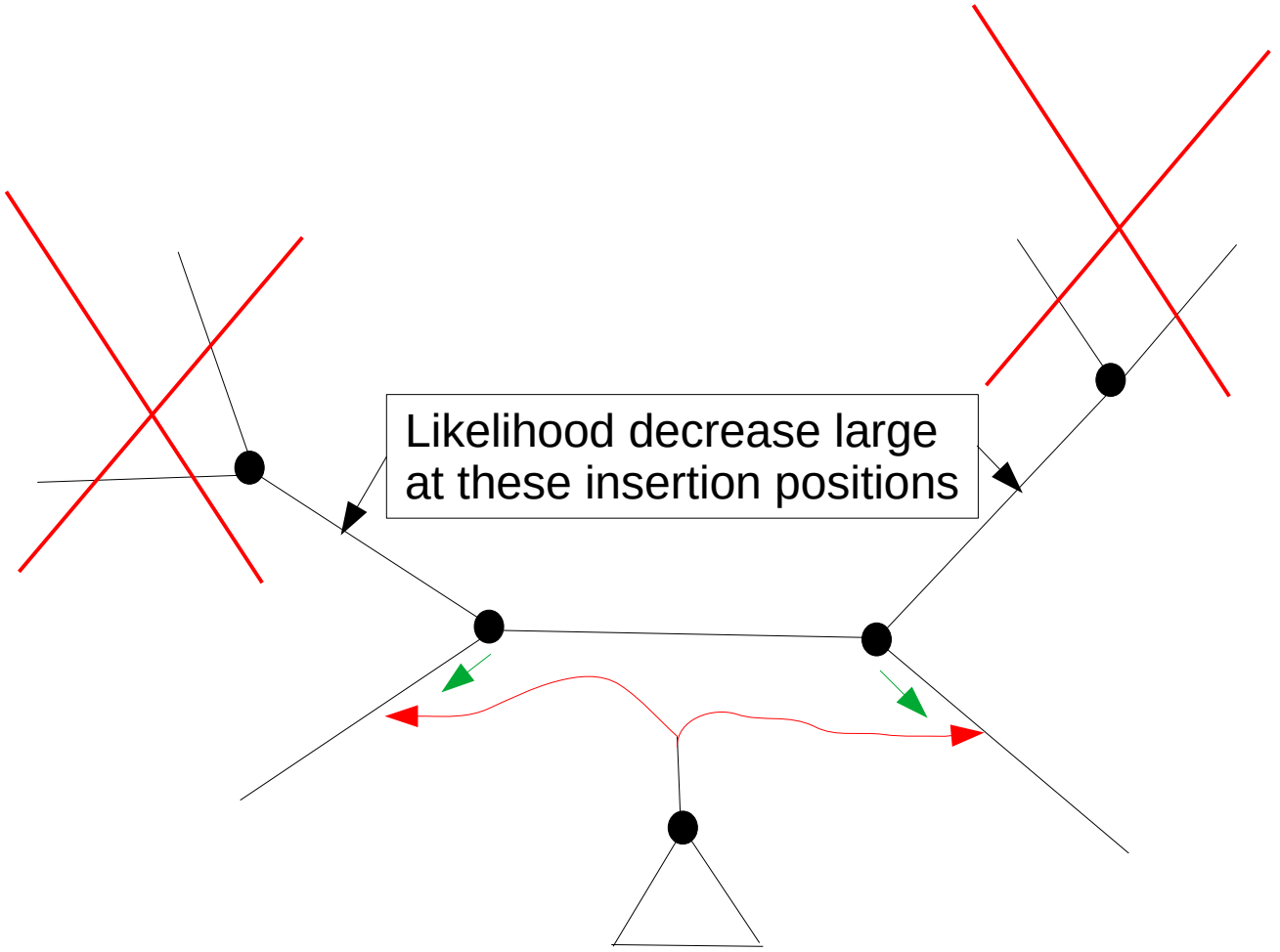
Why not parallelize by subtrees?

- Short answer: sequential dependencies & memory organization



1. CLV orientation! some CLVs need to be duplicated

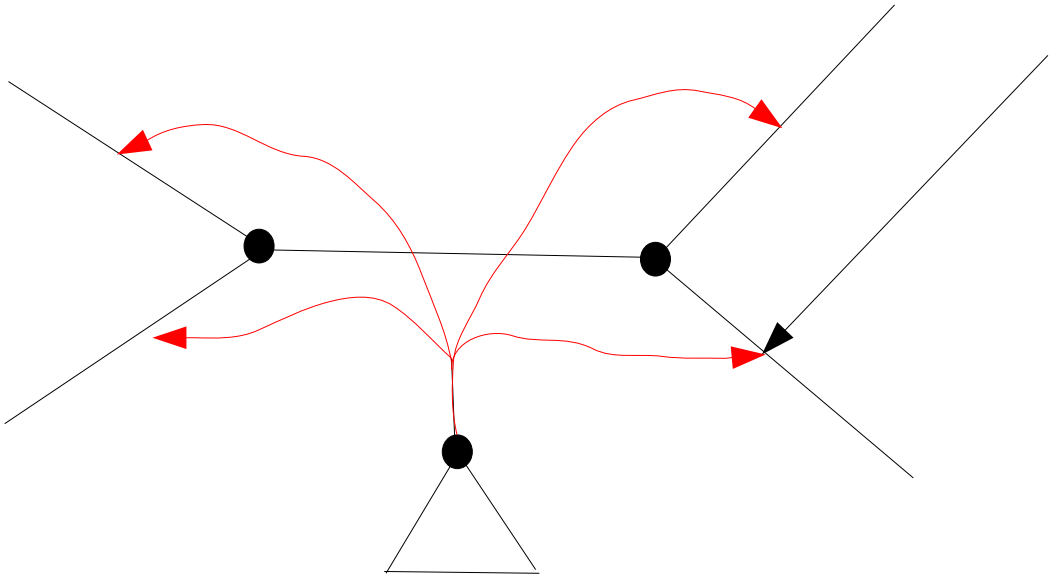
Why not parallelize by subtrees?



2. Heuristics, do not descend into non-promising subtrees

Why not parallelize by subtrees?

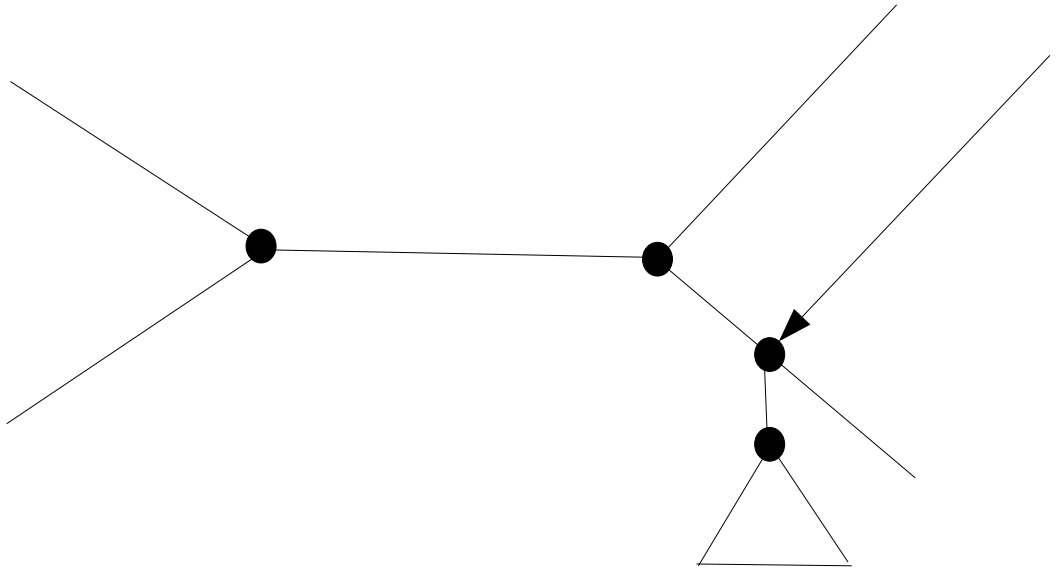
- Short answer: sequential dependencies & memory organization



3. If this insertion here Increases Likelihood keep tree immediately

Why not parallelize by subtrees?

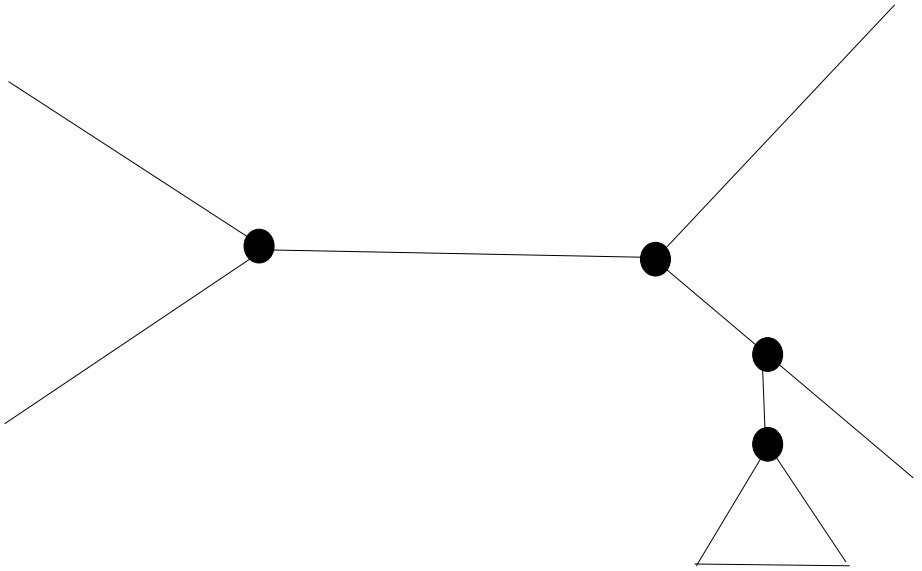
- Short answer: sequential dependencies & memory organization



If this insertion here Increases Likelihood keep tree immediately

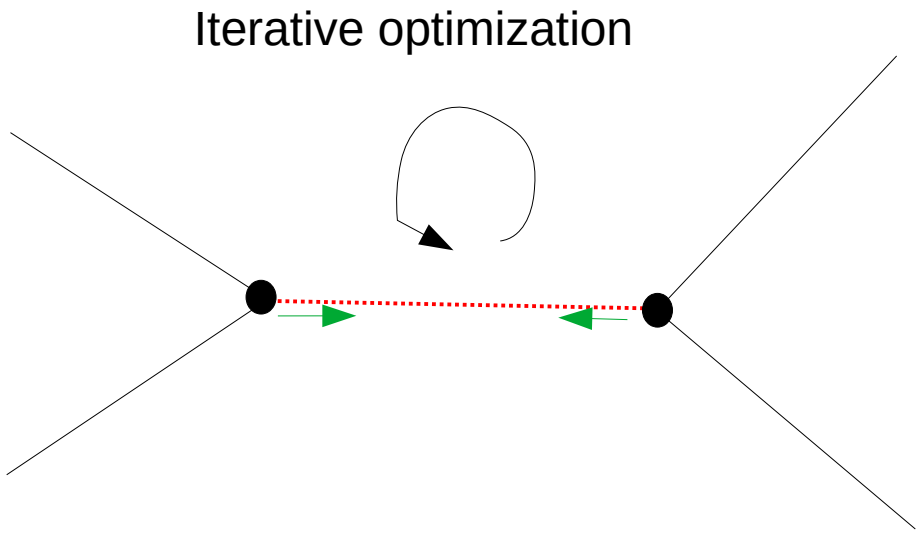
Why not parallelize by subtrees?

- Also: the subtree re-insertion traversal is built such that on average only 2-3 CLVs need to be updated as we move from one subtree insertion position to the next



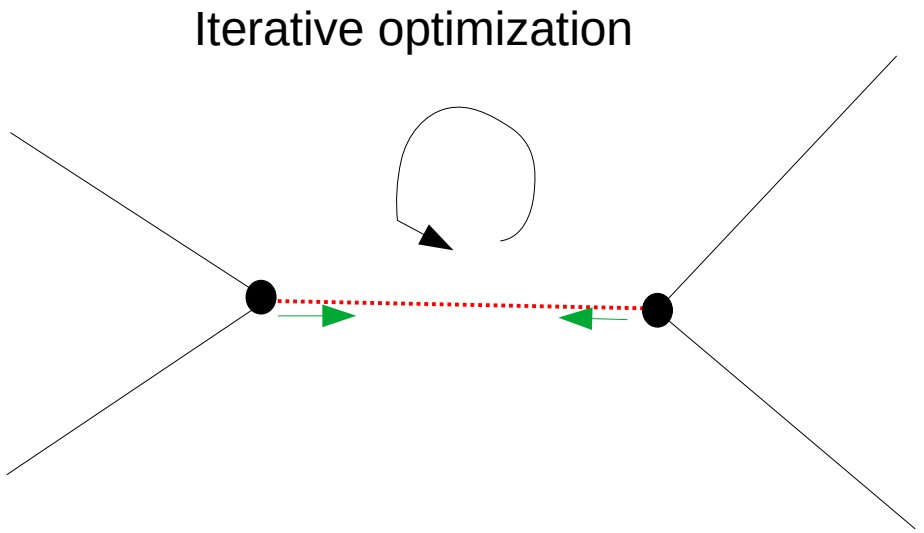
Other operations

- There is also branch length optimization that takes up between 20-30% of total execution time and that is inherently sequential as we visit one branch at a time



Other operations

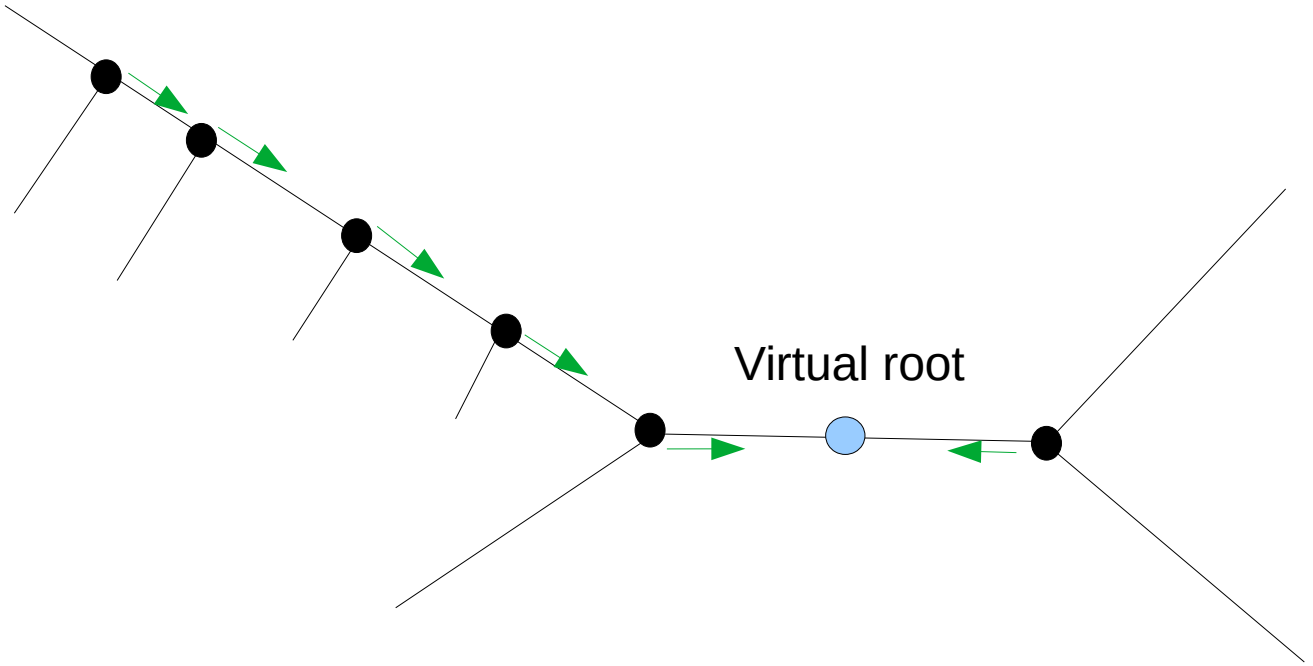
- There is also branch length optimization that takes up between 20-30% of total execution time and that is inherently sequential as we visit one branch at a time



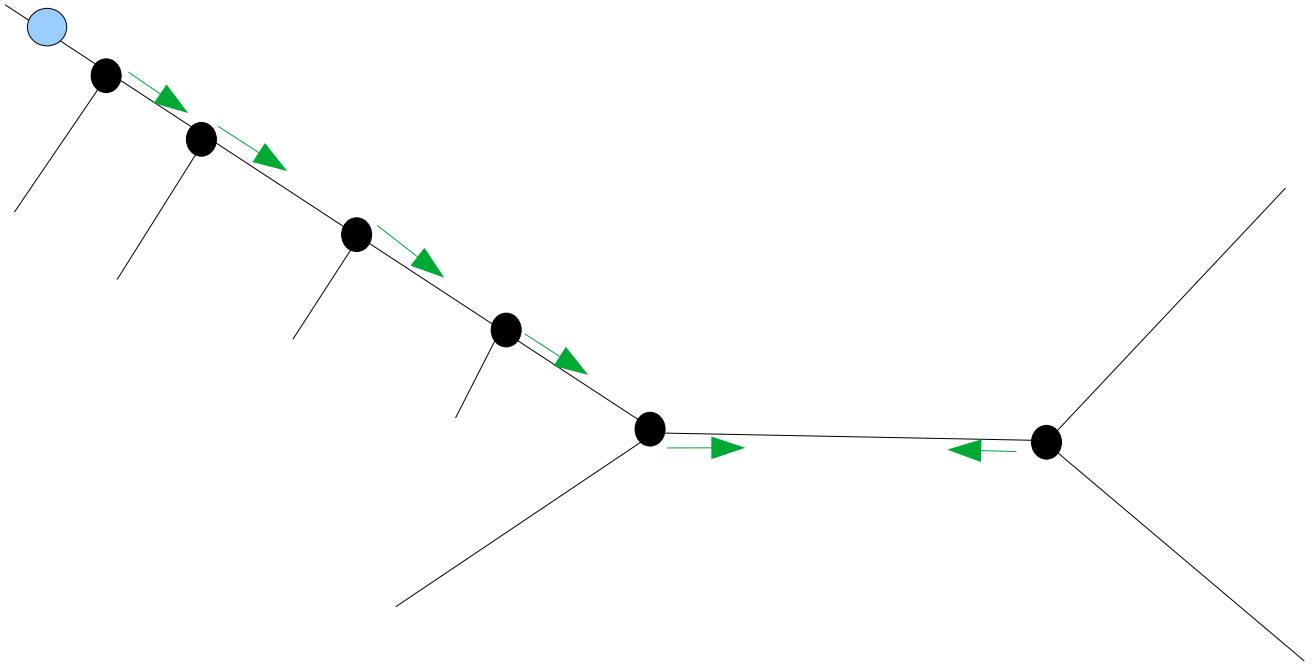
Different branches might require a different number of iterations to converge

Remote virtual roots

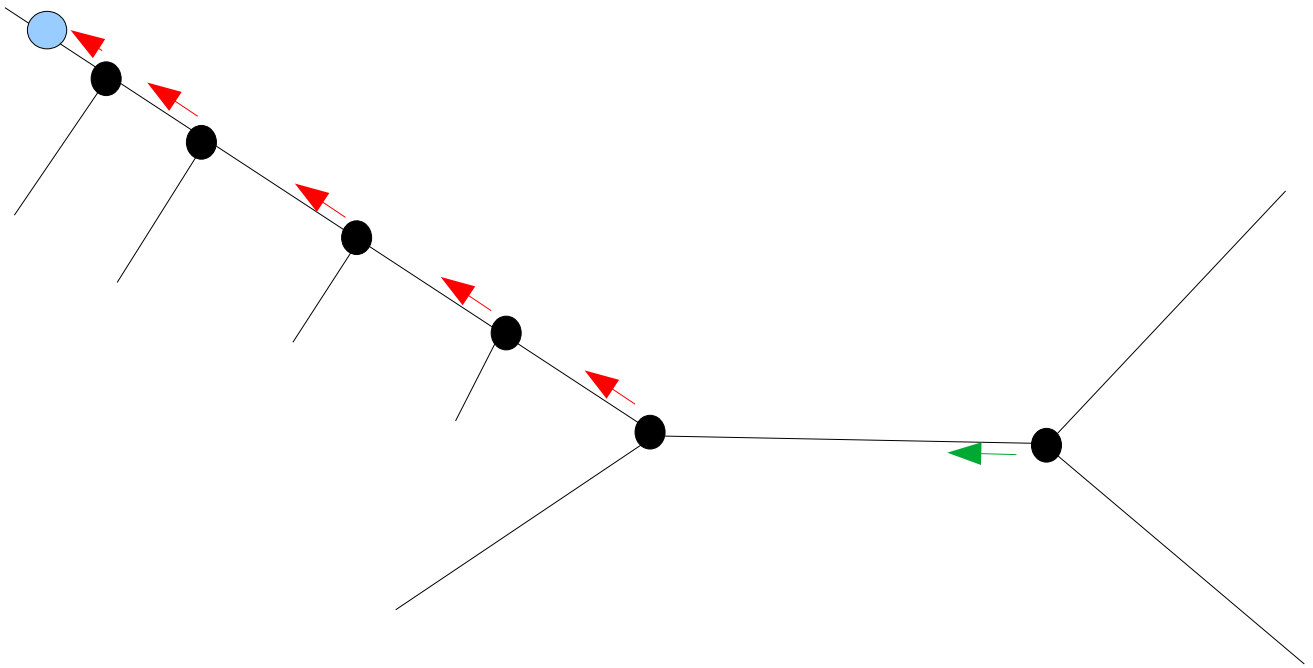
Question: If moving the root to an arbitrary edge, how can you determine the nodes you have to recalculate?



Remote virtual roots



Remote virtual roots



Where does rate heterogeneity come from

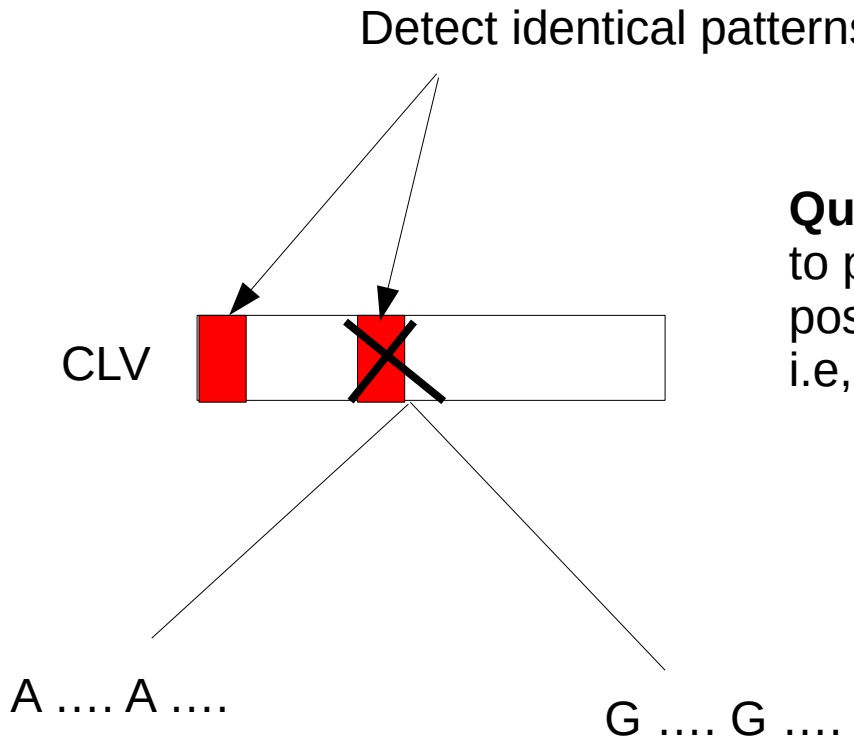
- In simple terms: by looking at a multiple sequence alignment of a gene some regions are highly conserved and some are not
- The highly conserved regions are essential for the gene's function (e.g., protein structure) the non-conserved regions are not
- This observation evidently comes from looking at alignments of extant (currently living) species
- Question: *If so, wouldn't that make the models inherently flawed because we'd be modeling probability of mutations to survive and not the actual probability for mutation?*

  → our models are always flawed :-)

  But not in this respect, as taking into account hidden/unobserved mutations is exactly the strength of the likelihood model.

  Remember the basic Markov property: $P(t+s)_{ij} = \Sigma\ P(t)_{ik} + P(s)_{kj}$ over **all** intermediate states *k*.

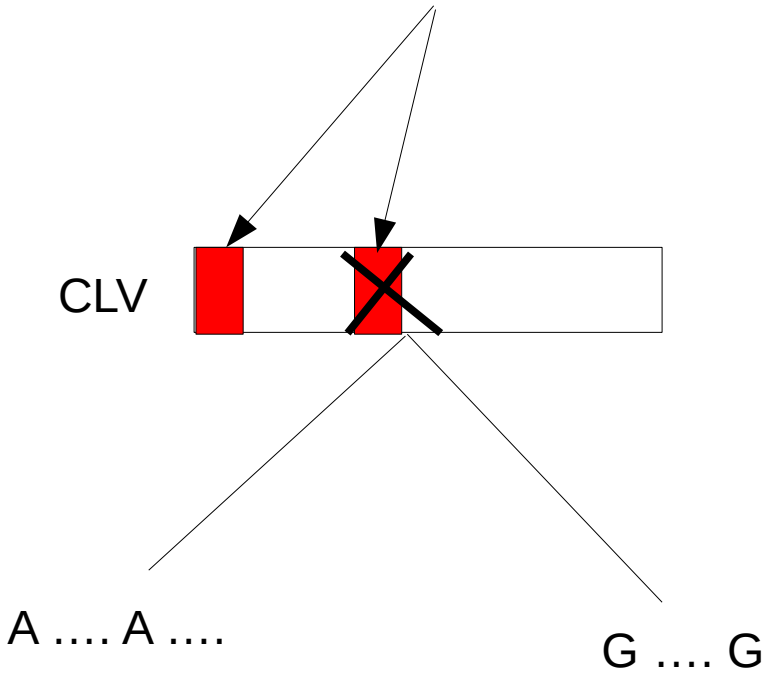
Repeating Patterns



Question: Wouldn't it be easier to pre-calculate and store all 4 X 4 possible patterns at the tips, i.e, *AA, AC, AG, AT, CA, CC, ….*

Repeating Patterns

Detect identical patterns and omit second computation



Question: Wouldn't it be easier to pre-calculate and store all 4 X 4 possible patterns at the tips, i.e, *AA, AC, AG, AT, CA, CC, ….*

Answer: not necessarily (i) we have 16 x 16 states due to Amibguous characters (ii) typically not all patterns will occur (iii) as we move higher up the tree there will be 16^k patterns where *k* is the number of leaves/sequences in the Subtree