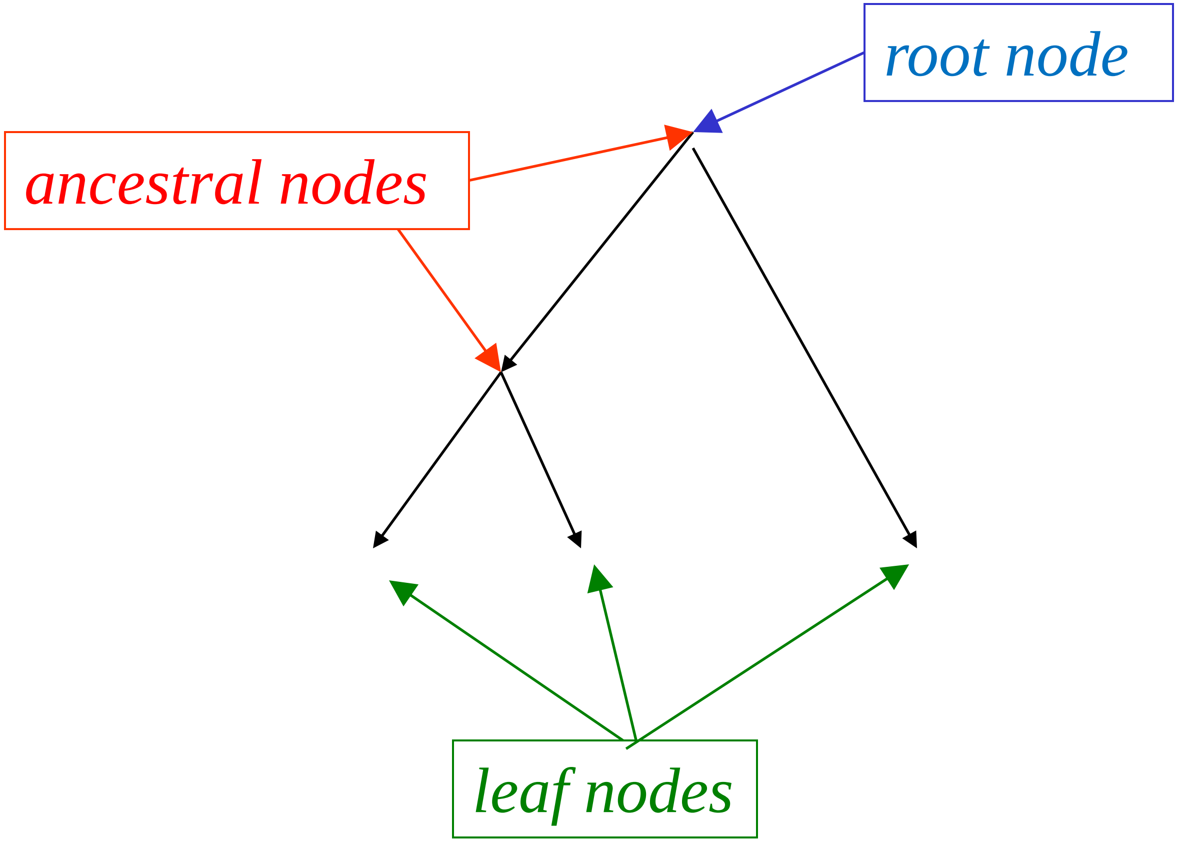
Lecture 16

- Evolutionary trees
- Tree-based probabilities for aligned sequences

Evolutionary trees

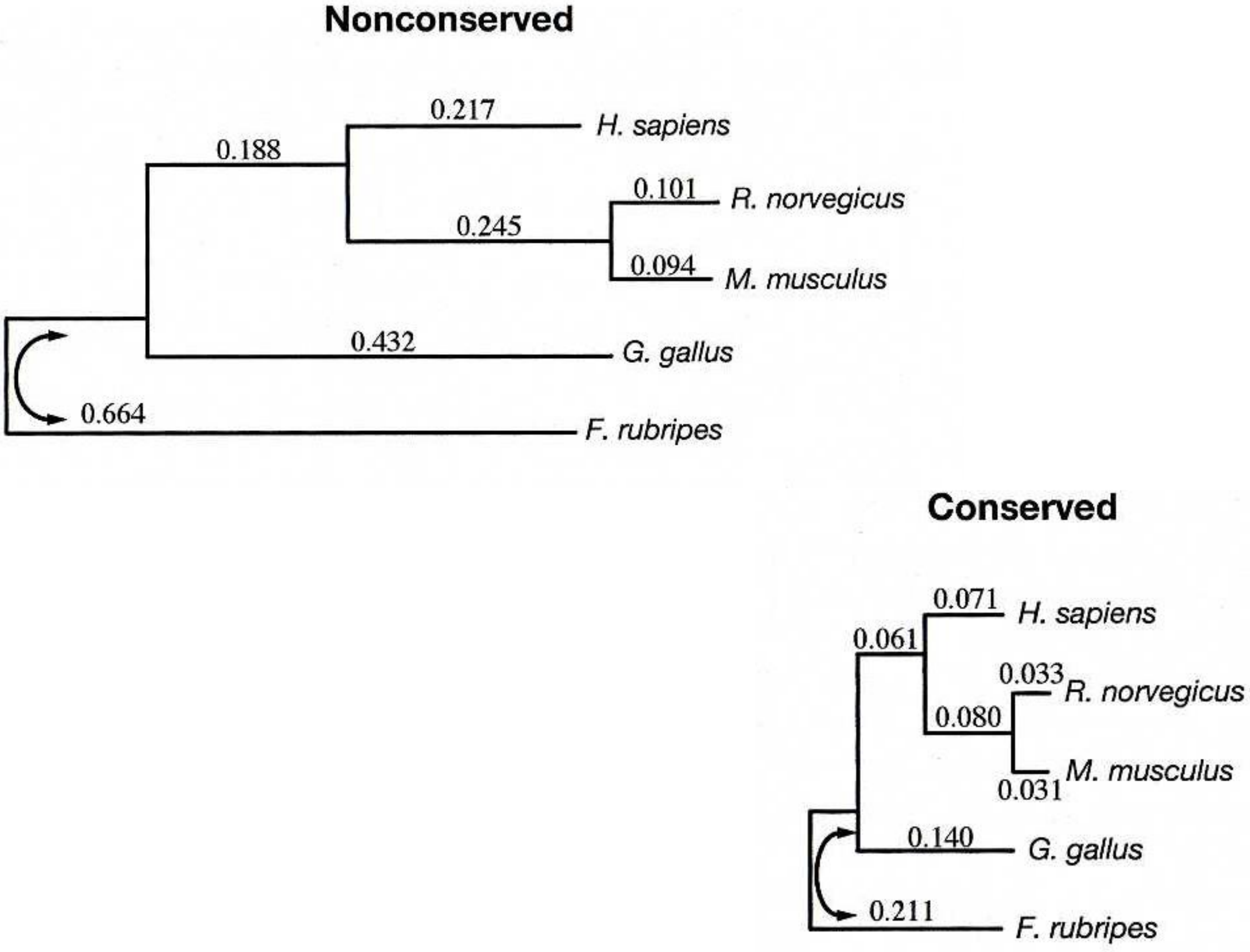
- Binary tree with
  - n_{leaf} ***leaf nodes*** (observed individuals)
  - n_{anc} ***ancestral nodes*** (unobserved)
- Each ancestral node has two descendants (‘left’ and ‘right’); leaves have none
- # edges:
  - # edge *starts* = $2\ n_{anc}$
  - # edge *ends* = $n_{leaf} + n_{anc} - 1$ (every node except root)
  - $\therefore 2\ n_{anc} = n_{leaf} + n_{anc} - 1$
  - $n_{anc} = n_{leaf} - 1$, # edges = $2\ n_{leaf} - 2$



root node

ancestral nodes

leaf nodes


Nonconserved
0.217
H. sapiens
0.188
0.101
R. norvegicus
0.245
0.094
M. musculus
0.432
G. gallus
0.664
F. rubripes
Conserved
0.071
H. sapiens
0.061
0.033
R. norvegicus
0.080
M. musculus
0.031
0.140
G. gallus
0.211
F. rubripes

- Want to compute *probabilities* of observed leaf sequences, given tree
  - Allows discriminating between possible trees
- Requires
  - considering possible sequences at ancestral nodes
    - # grows exponentially in both n_{anc} and sequence length !!
  - a probability model for change along edges

Mutational model for tree

- Will assume independent evolution at each sequence position
  - Doesn't allow for context effects (e.g. CpG hotspots!)
- Mutations along an edge *e*:

  $P_e(s \mid r)$ = prob a residue *r* at beginning of *e* is *s* at end
- 'Background' residue freqs at the root:

  $P_{root}(r)$

- Simplifying assumptions:
  - (for DNA) $P_e(s\wedge \mid r\wedge) = P_e(s \mid r)$
    - (^ = complementary nuc)
    - so each P_e has 6 independent params
  - A *single*, *reversible*, *infinitesimal* (~per small time unit) mutation model P_{inf} applies across entire tree
    - $P_e = (P_{inf}\,)^t$ where t = time along e
    - Reversibility implies root can’t be uniquely placed
    - This is model assumed by Siepel *et al*.

Probability calculations on tree

- Given:
    1. a set of observed residues at the leaves

       (a gap-free alignment column of the sequences)
    2. $\{P_e(s \mid r)\}$ and $\{P_{root}(s)\}$

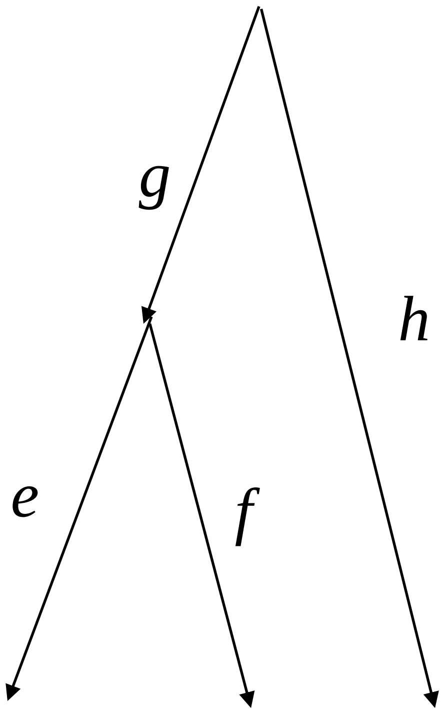
compute prob of observed residues

- Still exponentially many (in n_{anc}) possibilities for ancestral residues!
- But can use dynamic programming on a WDAG …

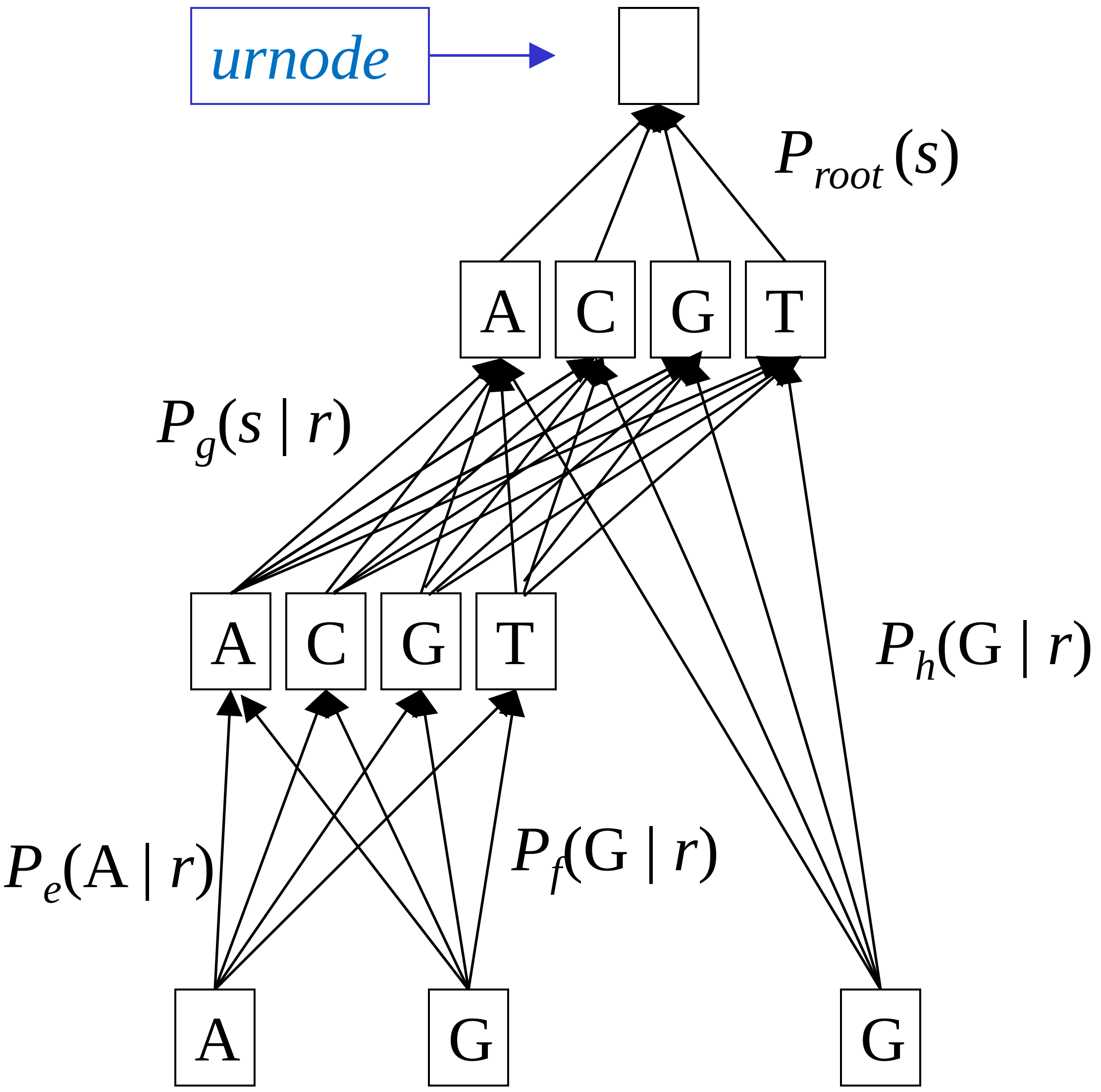
Evolut tree → WDAG

- Each ***ancestral node*** in tree becomes **4** nodes in WDAG
  - labelled with the 4 nucs
- ***leaf nodes*** remain unchanged
  - labelled with observed nuc
- Two nodes in WDAG are connected by an ***edge***

  if corresponding tree nodes are (but reverse direction)
  - weight = $P_e(s \mid r)$ where e = tree edge, r, s = node labels
- ‘urnode’
  - unlabelled
  - 4 edges coming from the 4 root nodes
  - weights = $P_{root}(s)$



g
h
e
f



urnode
$P_{root}(s)$
A
C
G
T
$P_g(s \mid r)$
A
C
G
T
$P_h(\mathrm{G} \mid r)$
$P_e(\mathrm{A} \mid r)$
$P_f(\mathrm{G} \mid r)$
A
G
G

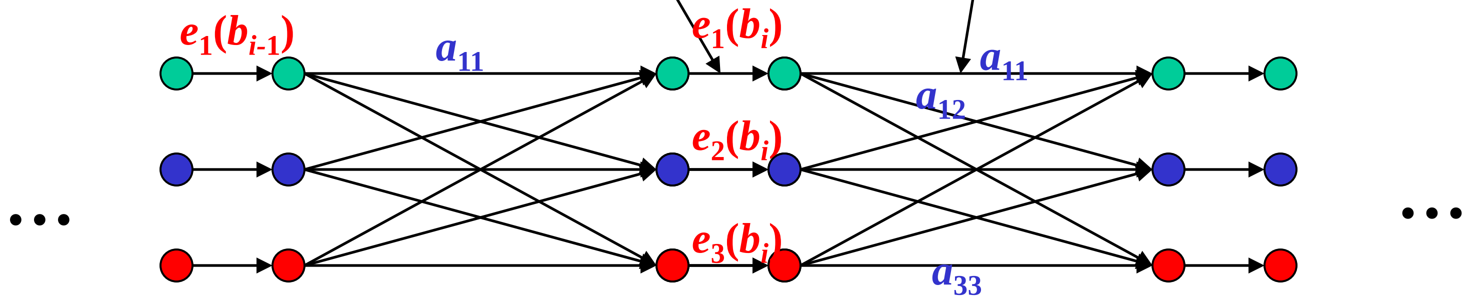
- Size of WDAG is linear in n_{leaf}
  - – # nodes: n_{leaf} + 4 n_{anc} + 1
  - – # edges: 4 n_{leaf} + 16 (n_{anc} – 1) + 4

- Edges in tree point *down*; in WDAG, *up*
  - – so WDAG ‘parents’ are *below*

cf. WDAG for 3-state HMM length n sequence (lecture 14)

weights are emission probabilities $e_k(b_i)$ for i^{th} residue b_i

weights are transition probabilities a_{kl}



$b_{i\text{-}1}$
position *i*-1

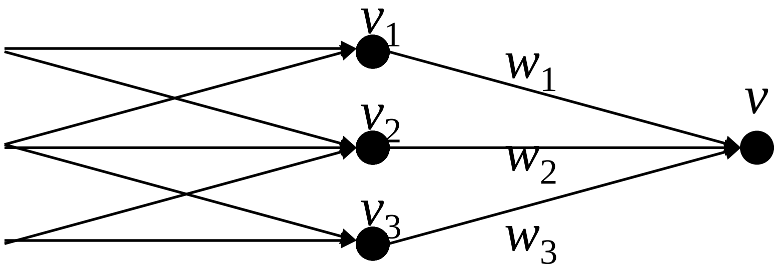
b_i
position *i*

b_{i+1}
position *i*+1

Prob calcs in HMMs (lecture 15):

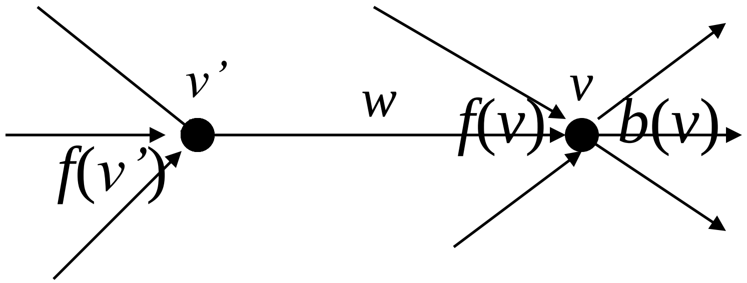
For each vertex v, let $f(v) = \sum_{\text{paths } p \text{ ending at } v} \text{weight}(p)$, where weight($p$) = *product* of edge weights in p. Only consider paths starting at ‘begin’ node.

Compute $f(v)$ by dynam. prog: $f(v) = \sum_i w_i\, f(v_i)$, where v_i ranges over the parents of v, and w_i = weight of the edge from v_i to v.



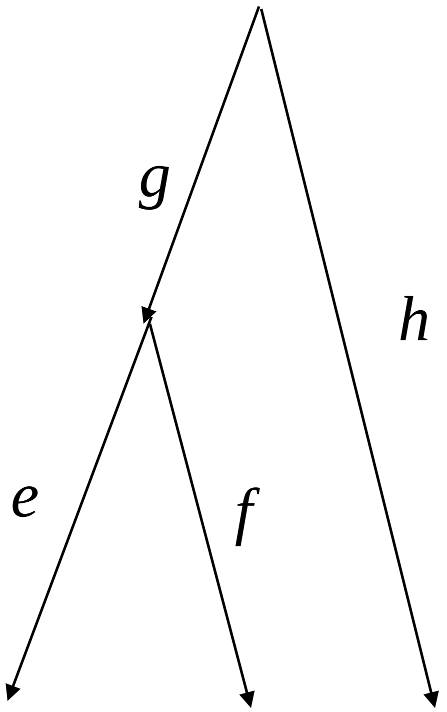
Similarly for $b(v) = \sum_{p \text{ beginning at } v} \text{weight}(p)$

The paths *beginning* at v are the ones *ending* at v in the *reverse* (*or inverted*) *graph*

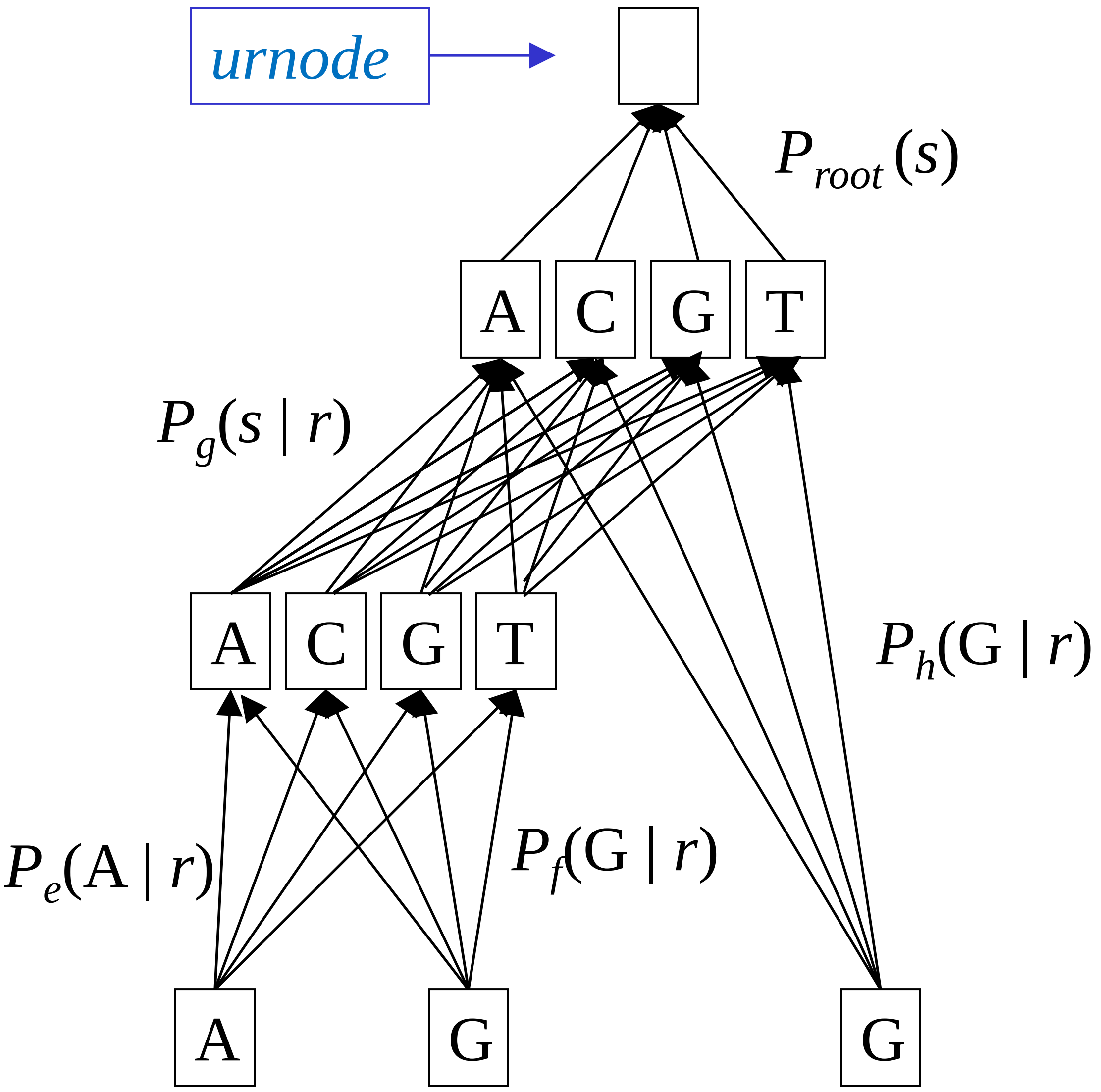


$f(v)b(v)$ = sum of the path weights of all paths *through v*.

$f(v')wb(v)$ = sum of the path weights of all paths *through the edge (v',v)*



g
h
e
f



urnode
$P_{root}(s)$
A C G T
$P_g(s \mid r)$
A C G T
$P_h(\mathrm{G} \mid r)$
$P_e(\mathrm{A} \mid r)$
$P_f(\mathrm{G} \mid r)$
A
G
G

- Compute overall *probability* of leaf residues (nucleotides) by *dynamic programming* on WDAG:

- Let, for each node v, $f(v)$ = prob of leaf nucs *below v* (i.e tree-descendants, or WDAG-ancestors, of v), given v's nuc

  $f_{left}(v)$ = prob of leaf nucs *below* and to *left*

  $f_{right}(v)$ = prob of leaf nucs *below* and to *right*

  then $f(v) = f_{left}(v)\, f_{right}(v)$

- Compute these values node-by-node, visiting (WDAG-)parents before children:
  - *starting* at leaf nodes (setting $f(v) = 1$), *ending* at urnode

$f_{left}(v) = \sum_{left-u} w(u,v) f(u)$ where

  - u ranges over parent nodes to the left
  - $w(u,v)$ = weight on edge from u to v

    (= mutation prob from v to u)

Similarly for $f_{right}(v)$

$f(v) = f_{left}(v)\, f_{right}(v)$

  - For v = urnode, view *all* parents as being to ‘left’ and $f(v) = f_{left}(v)$

- f(urnode) = probability of the observed leaf nucs

- a ‘forward-backward’ calc gives posterior prob of having
  - a particular nuc at an ancestral node, or
  - a particular mutational change along an edge
- can use these as *fractional counts* to estimate *P*’s (EM algorithm)

Probability models & alignments

- Getting the probability model P_e requires a multiple alignment

- But optimal (LLR) scoring for alignment uses P_e :
  - log((prob of col | P_e model) / (prob of col | background))

- Find P_e , alignment jointly & iteratively (Sankoff):
  - crude alignment → P_e → scores → better alignment etc