06-13 The posterior decoding computes the most probable state at any position given an observation (it is taken over all possible sequences that fit the observation).

The Viterby algorithm computes the most probable state sequence for the observation.

(Note that the sequence of most probable states is usually not equal to the most probable state sequence.)

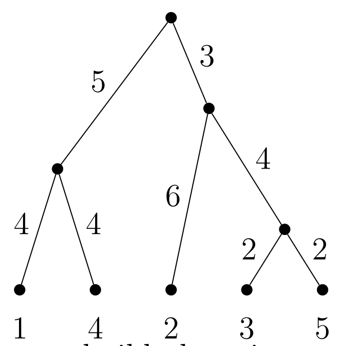
05-13 The Viterbi algorithm is a form of dynamic programming. For each state $p \in Q$ of the model, and each position $i \leq n$ of the observation $x_1 \ldots x_n$, compute the maximal probability $v_q(i)$ of a state sequence ending in p with the observation upto position i.

In fact, $v_q(i) = \max_{p \in Q} v_p(i-1) t_{pq} e_{qx_i}$, where t_{pq} are the transition probabilities, and $e_{q\sigma}$ the emission probabilities.

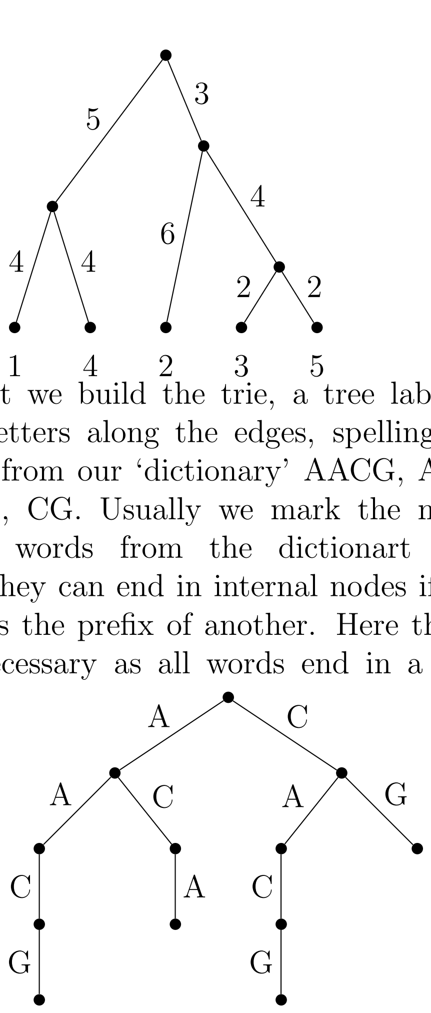
05-14 Uniform clock implies all leaves are at the same distance from the root. The maximal distance is between two nodes that are only connected via the root. Here that distance is 18, so all leaves are 9 below the root.

As 1 & 4 have maximal distance to 2,3 & 5, this means that the tree can be divided at the root into two subtrees, with 1,4 in one and 2,3,5 in the other subtree. Now continue recursively. The distance of 1&4 is 8, which implies they are distance 4 below their common ancestor. This ancestor then is 5 below the root.

Now consider the matrix restricted to 2,3,5. Max distance now is 12, between 2 and 3 & 5. Thus 2 is 6 below its ancestor, which is 3 below the root; 3 & 5 are 4 apart, so 2 below their ancestor, which leaves 4 for the distance to the node where the branch to 2 was.



05-15 First we build the trie, a tree labelled with letters along the edges, spelling the words from our 'dictionary' AACG, ACA, CACG, CG. Usually we mark the nodes where words from the dictionart end, since they can end in internal nodes if one word is the prefix of another. Here this is not necessary as all words end in a leaf.



Now determine failure links. Each node links to the longest prefix that is a suffix on the path to the present node. For instance, the first leaf 'AACG' has suffix ACG (which is not in the tree) and CG which is in the tree, so we have found the failure link. Links are computed as in the Knuth-Morris-Pratt algorithm for single strings. Follow the failure links starting at the parent, until the right label is found (or when we move out of the tree).

Links of A and C are to the root λ.
Link of AA, we look for the last letter A: follow the link at A to λ, there we find A. The link of AA points to A.

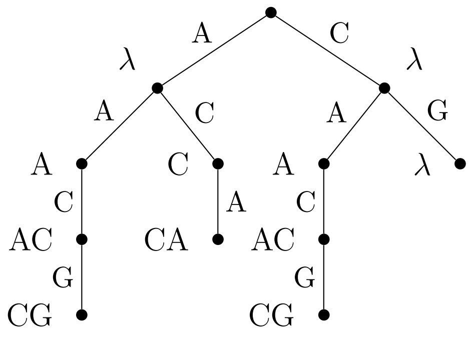
Similarly the link at AC points to C, and that of CA points to A.

For CG: the link of parent C points to the root, where there is no G. We cannot go up, so CG points to the root.

For AAC: link of parent AA points to A, where there is a C. So AAC points to AC.

For ACA: link of parent AC points to C, where there is a A. So ACA points to CA. Similarly CAC points to AC. etc.

Deawing the pointers as upward edges takes some effort, so we indicate them using labels.


A
C
λ
λ
A
C
A
G
A
C
A
λ
C
A
C
AC
CA
AC
G
G
CG
CG