Bioinformatics

Richard Bonneau

Lecture 6: Probabilistic phylogenetic trees.

NEW YORK UNIVERSITY
CENTER FOR COMPARATIVE
FUNCTIONAL GENOMICS

COURANT
INSTITUTE

Associated reading.

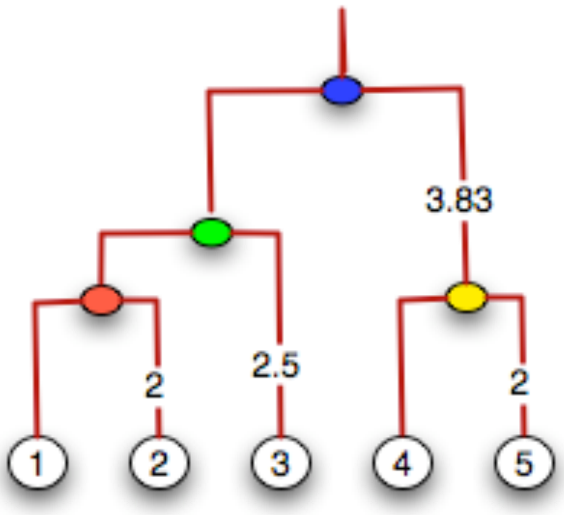
- Ch. 8 BSA
- MAIN:
  Large punctual contribution of speciation to evolutionary divergence. Science 314:2006, p. 119
- Optional:
  Branch and bound algorithms to determine minimal evolutionary trees. Hendy + Penny. Mathamatical Biosciences 59:277-290(1982)

Making trees from pairwise distances

d	1:	2:	3:	4:	5:
1:	0				
2:	2	0			
3:	2	3	0		
4:	5	4	4	0	
5:	4	4	2	2	0

```
1: AGCTTC-TA
2: ACGTTCTTA
3: AGCTTATTA
4: TCCTATTTA
5: TCCTTATTA
```

Where distance is number of mismatches



Neighbor joining:

We still assume additivity, we still use a deterministic joining algorithm, but we redefine distance and the algorithms slightly to better deal with variable branch lengths.

$$d_{ij} = -3/4 \log\left(1 - 4f/3\right)$$

$$D_{ij} = d_{ij} - (r_i + r_j)$$

We calc a distance D, where d is corrected by mean path to other nodes, r. (where L is number of leaves)

$$r_i = \frac{1}{|L| - 2} \sum_{k \in L} d_{ik}$$

$$d_{k\bullet} = \frac{1}{2}(d_{i\bullet} + d_{j\bullet} - d_{ij})$$

Algorithm:

0. leaf nodes -> L, we'll grow tree from this set, L
1. Pick argmin$_{ij}$(D$_{ij}$) and make node k joining i and j
2. Calc distance from k to all other nodes
3. Add k to growing tree
4. Remove i and j from node list (now they are represented by k
5. Rinse, lather, repeat.

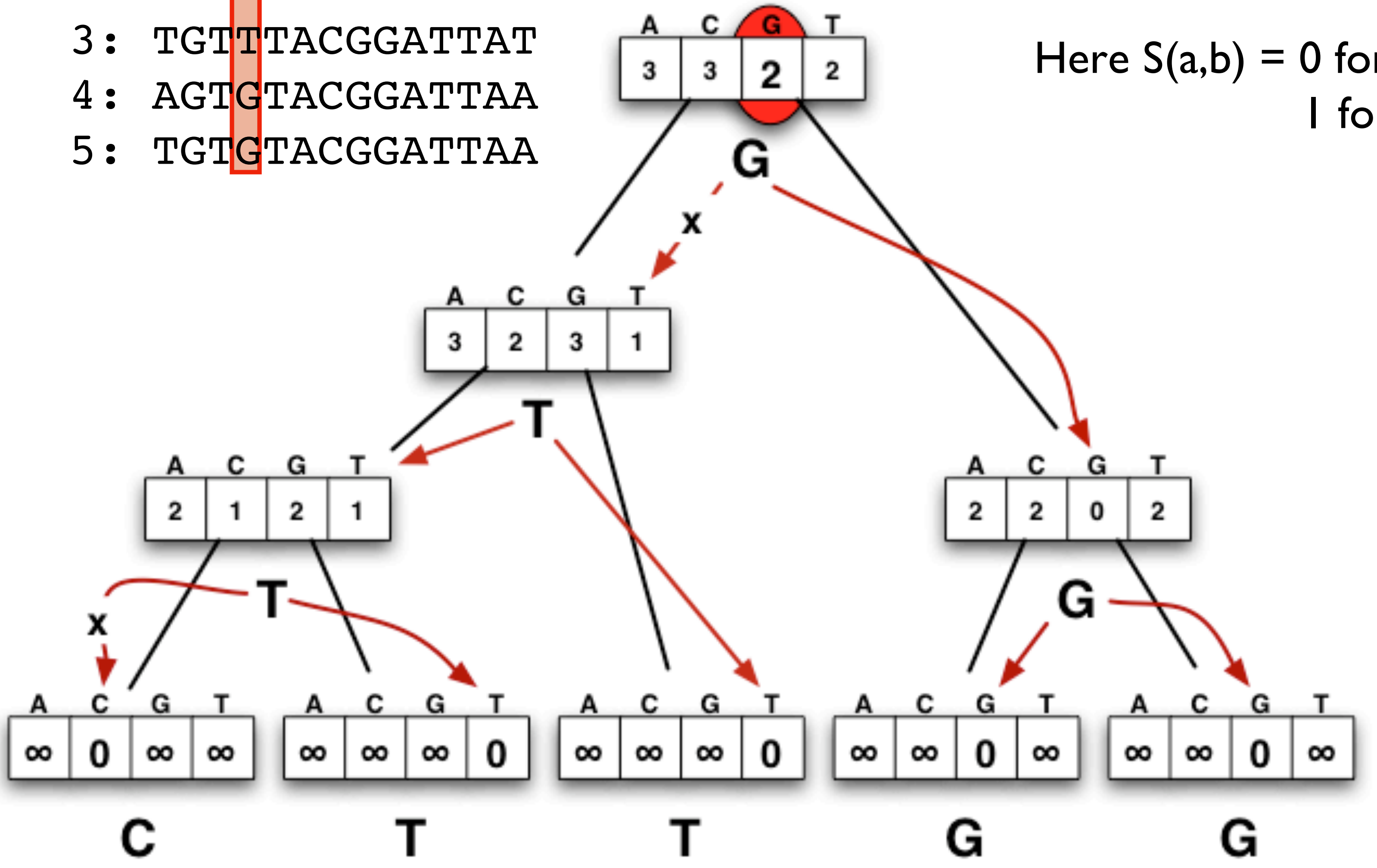
parsimony

Evaluate cost of tree: <u>Given tree + Given alignment.</u>

```
1: AGTCTACGGATTAT
2: AGTTTACGGATTAT
3: TGTTTACGGATTAT
4: AGTGTACGGATTAA
5: TGTGTACGGATTAA
```

1. Set $S_k(a) = 0$ for $a = x$, infinity otherwise for leaf nodes.
2. If not at leaf node (internal vertex):
   $$S_k(a) = \min_b(S_i(b) + S(a,b)) + \min_b(S_j(b) + S(a,b))$$

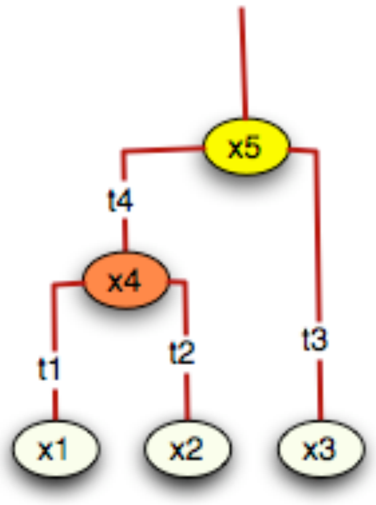
Here $S(a,b) = 0$ for match, 1 for missmatch



P(tree|data)

Evaluate cost of tree: Given tree + Given alignment. With variable branch lengths. We aim to maximize P(tree|Data)

```
1: AGTCTACGGATTAT
2: AGTTTACGGATTAT
3: TGTTTACGGATTAT
```



We want to calc:
$P(x^{\bullet}|T,t_{\bullet})$

Where :
$x^{\bullet}$ = sequences
T = the tree topology
$t_{\bullet}$ = the branch lengths

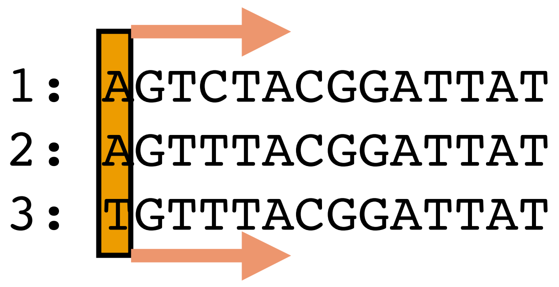
$$P(x^1,x^2,x^3,x^4,x^5 \mid T,t_{\bullet}) = P(x^1 \mid x^4,t_1)P(x^2 \mid x^4,t_2)P(x^3 \mid x^5,t_3) P(x^4 \mid x^5,t_4)P(x^5)$$

How do we calc:

$$P(x \mid y,t)$$

S(t)

Positional independence:



```
1: AGTCTACGGATTAT
2: AGTTTACGGATTAT
3: TGTTTACGGATTAT
```

Product over positions to calc P(x|y,t):

$$P(x \mid y,t) = \prod_i P(x_i \mid y_i, t)$$

Multiplicative with respect to time:

$$S(t_1 + t_2) = S(t_1)S(t_2)$$

$$P(a \mid b, t_1)P(b \mid c, t_2) = P(a \mid c, t_1 + t_2)$$

S(t) … rates

All substitutions equal

$$R = \begin{pmatrix} -3\alpha & \alpha & \alpha & \alpha \\ \alpha & -3\alpha & \alpha & \alpha \\ \alpha & \alpha & -3\alpha & \alpha \\ \alpha & \alpha & \alpha & -3\alpha \end{pmatrix}$$

We need to defign a substitution matrix that is time dependant.

We define expected rates as matrix R

Transitions and Transversions at different rates

$$R = \begin{pmatrix} -2\beta-\alpha & \alpha & \alpha & \beta \\ \alpha & -2\beta-\alpha & \beta & \alpha \\ \alpha & \beta & -2\beta-\alpha & \alpha \\ \beta & \alpha & \alpha & -2\beta-\alpha \end{pmatrix}$$

S(t)

For small t

$$S(t) \simeq (I + R\varepsilon) =$$

$$\begin{pmatrix} 1-3\alpha\varepsilon & \alpha\varepsilon & \alpha\varepsilon & \alpha\varepsilon \\ \alpha\varepsilon & 1-3\alpha\varepsilon & \alpha\varepsilon & \alpha\varepsilon \\ \alpha\varepsilon & \alpha\varepsilon & 1-3\alpha\varepsilon & \alpha\varepsilon \\ \alpha\varepsilon & \alpha\varepsilon & \alpha\varepsilon & 1-3\alpha\varepsilon \end{pmatrix}$$

S(t)

Solving for s(t)

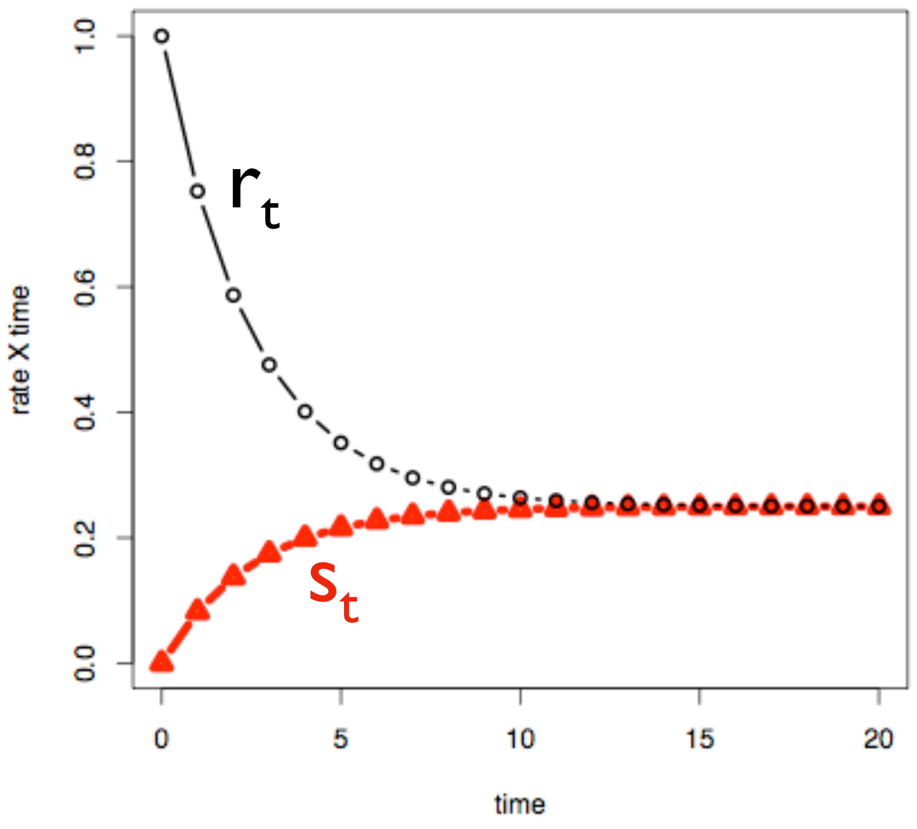
$$\frac{\partial(S(t))}{\partial t} = S(t)R$$

$$S(t) = \begin{pmatrix} r_t & s_t & s_t & s_t \\ s_t & r_t & s_t & s_t \\ s_t & s_t & r_t & s_t \\ s_t & s_t & s_t & r_t \end{pmatrix}$$

$$r_t = \frac{1}{4}(1 + 3e^{-4\alpha t})$$

$$s_t = \frac{1}{4}(1 + e^{-4\alpha t})$$

As t->∞, $r_t = s_t = 0.25$



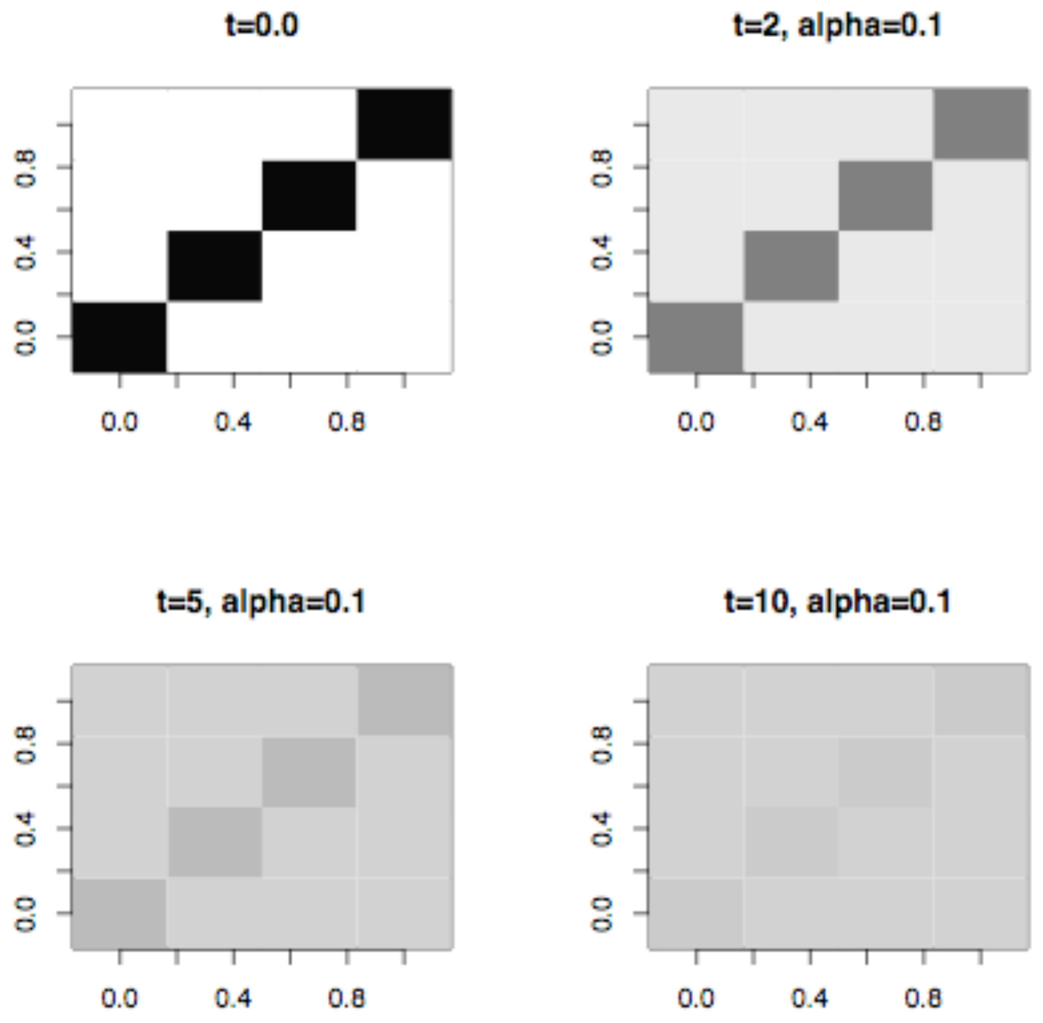
S(t)

Solving for s(t)

$$S(t) = \begin{pmatrix} r_t & s_t & s_t & s_t \\ s_t & r_t & s_t & s_t \\ s_t & s_t & r_t & s_t \\ s_t & s_t & s_t & r_t \end{pmatrix}$$

$$r_t = \frac{1}{4}(1 + 3e^{-4\alpha t})$$

$$s_t = \frac{1}{4}(1 + e^{-4\alpha t})$$



Two sequences

The P(x|T,t) is root invariant for two gene case.

$$P(x_u^1, x_u^2, a \mid T, t_1, t_2) =$$

$$q_a P(x_u^1 \mid a, t_1) P(x_u^2 \mid a, t_2)$$

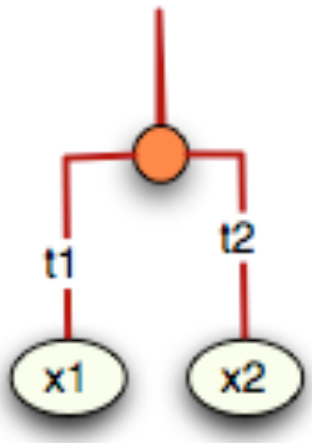
$$P(x_u^1, x_u^2 \mid T, t_1, t_2) = \sum_a q_a P(x_u^1 \mid a, t_1) P(x_u^2 \mid a, t_2)$$

$$P(x^1, x^2 \mid T, t_1, t_2) = \prod_u P(x_u^1, x_u^2 \mid T, t_1, t_2)$$

$$P(x^1 = C, x^2 = C \mid T, t_1, t_2) = q_C r_{t_1} r_{t_2} + q_A s_{t_1} s_{t_2} + q_G s_{t_1} s_{t_2} + q_T s_{t_1} s_{t_2}$$

$$P(C, C \mid T, t_1, t_2) = \frac{1}{16}(1 + 3e^{-4\alpha\underline{(t_1+t_2)}})$$

$$P(C, G \mid T, t_1, t_2) = \frac{1}{16}(1 - e^{-4\alpha\underline{(t_1+t_2)}})$$

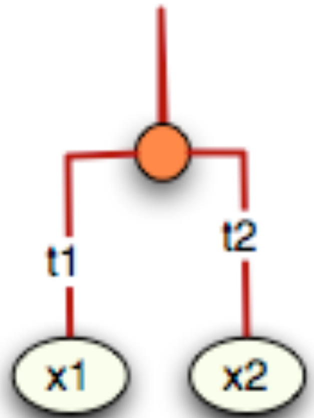


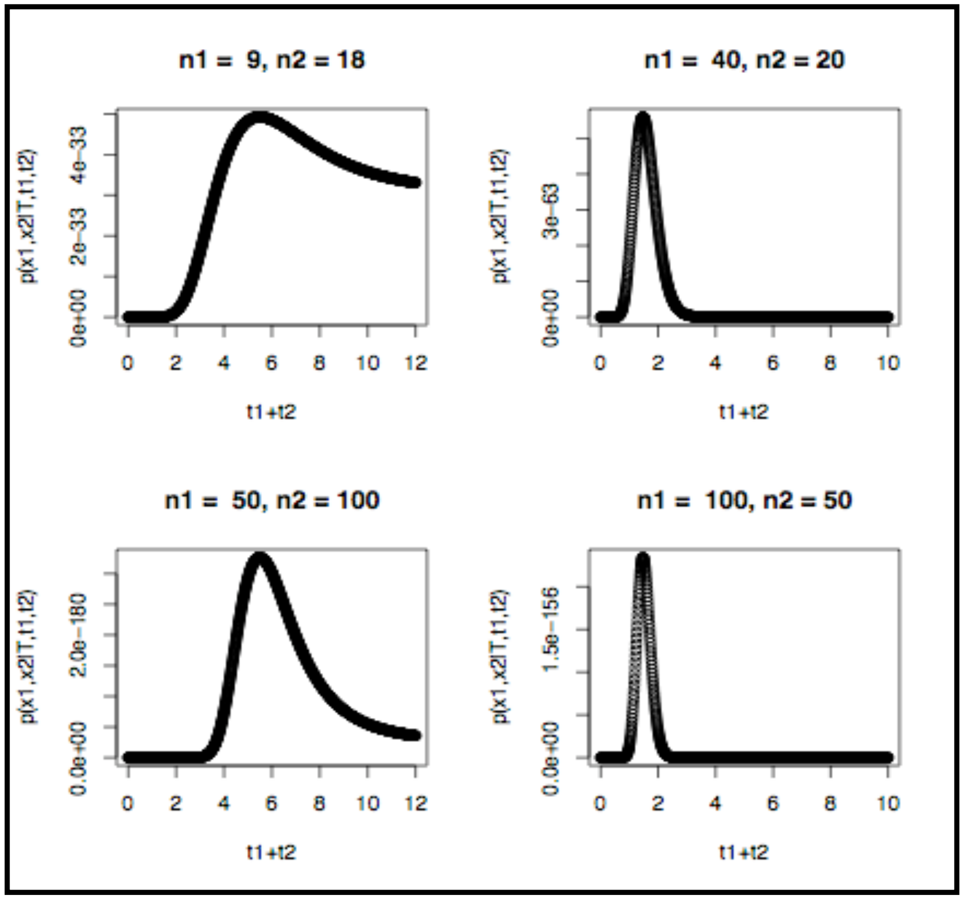
Two sequences

$$P(C,C \mid T,t_1,t_2) = \frac{1}{16}(1+3e^{-4\alpha(t_1+t_2)})$$

$$P(C,G \mid T,t_1,t_2) = \frac{1}{16}(1-e^{-4\alpha(t_1+t_2)})$$

$$P(x_1,x_2 \mid T,t_1,t_2) = \frac{1}{16^{n_1+n_2}}(1+3e^{-4\alpha(t_1+t_2)})^{n_1}(1-e^{-4\alpha(t_1+t_2)})^{n_2}$$





The P(x|T,t) is root invariant for two gene case. If we carry the product over all sequence positions and make n_1 = number of matches and n_2 = number of mismatches.

We show P as a function of t1+t2 (alpha = 0.01) for various values of n1 and n2.

n sequences

$$P(x_u^1,\ldots,x_u^n \mid T,t_\bullet) =$$

$$\sum_{a^{n+1},a^{n+2},\ldots,a^{2n-1}} q_{a^{2n-1}} \prod_{i=n+1}^{2n-2} P(a^i \mid a^{\alpha(i)},t_i) \prod_{i=1}^{n} P(x_u^i \mid a^{\alpha(i)},t_i)$$

Given T and t.
α(i) is immediate ancestor to node
i.
X's represent sequence positions

a's represent sequence possibilities at
internal nodes.

The sum is over all possible assignments
of a at each non-leaf node… this could
mean a big computation per evaluation
of each T,t over X …

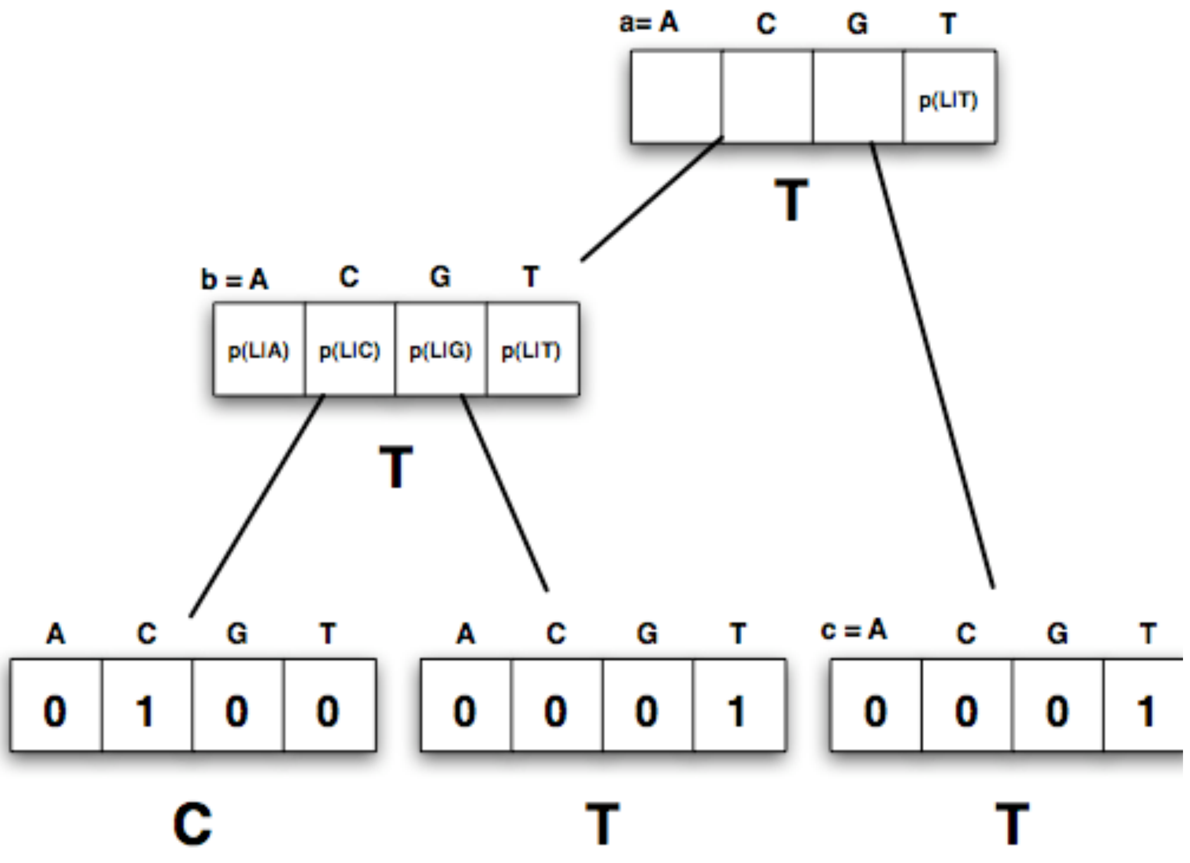
n sequences

Felsenstein's Algorithm [1981]

$$P(x_u^1,\ldots,x_u^n \mid T,t_\bullet) =$$

$$\sum_{a^{n+1},a^{n+2},\ldots,a^{2n-1}} q_{a^{2n-1}} \prod_{i=n+1}^{2n-2} P(a^i \mid a^{\alpha(i)},t_i) \prod_{i=1}^{n} P(x_u^i \mid a^{\alpha(i)},t_i)$$

```
$init:$
    $k = 2n-1$
$recursion(P(L_k \mid a)):$
    $if\,(k = leaf):$
    $P(L_k \mid a = x_u^k) = 1; P(L_k \mid a \neq x_u^k) = 0$
    $if\,(k > n):$
    $P(L_k \mid a) =$
    $\sum_{b,c} P(b \mid a,t_i)P(L_i \mid b)P(c \mid a,t_j)P(L_j \mid c)$
$term:$
    $l_u = P(x_u^\bullet \mid T,t_\bullet) = \sum_a P(L_{2n-1} \mid a)q_a$
```



Finding most probable trees

For small trees numerically solve for maximum likelihood tree

Or, maximum likelihood algorithm proposed by Felsenstein.

Conjugate gradient.

Finding most probable trees

For small trees numerically solve for maximum likelihood tree

Or, maximum likelihood algorithm proposed by Felsenstein.

Conjugate gradient.

We can also use Monte Carlo to sample from

$$P(T,t_\bullet \mid x^\bullet) = \frac{P(x^\bullet \mid T,t_\bullet)P(T,t_\bullet)}{P(x^\bullet)}$$

$$P_1 = P(T,t_\bullet \mid x^\bullet)$$

$$P_2 = P(\tilde{T},\tilde{t}_\bullet \mid x^\bullet)$$

If $P_2 \leq P_1$ accept move
If $P_2 > P_1$ accept move with $P \sim P_2/P_1$

Moves are defined by a so-called proposal distribution. Possible moves to change one tree into another:

1. Change node height
2. Reordering leaves / branch switching

o Still a very difficult search.

Parsimony and Felsenstein algorithm

We can relate the weighted parsimony algorithm to the ML algorithm of Felesnstein.

Score from parsimony can be related to P:

$$S(a,b) = -\log P(b \mid a)$$

We see that parsimony uses:

$$\min(S(b) + S(a,b)) \approx \max(P(b)P(b \mid a))$$

While maximum likelihood algorithm uses:

$$\sum_b P(b)P(b \mid a)$$

Thus we can think of the weighted parsimony algorithm as a Viterbi approximation of the of the ML result with fixed branch length given the tree.

<u>Problems:</u>

No branch length optimization, so several cases where parsimony does quite poorly.

Neighbor joining -> Maximum Likelihood

…If prob model is correct.

Prob methods/models let us:
-assess tree
-generate ensembles of plausible trees
-use priors

Neighbor joining could be used to generate plausible starting trees

With multiplicative and reversible P's for substitutions we can show neighbor joining correctly reconstructs tree:

$$P(a^1 | a^8, t_1 + t_6) = \sum_{a_6} P(a^1 | a^6, t_1) P(a^6 | a^8, t_6)$$

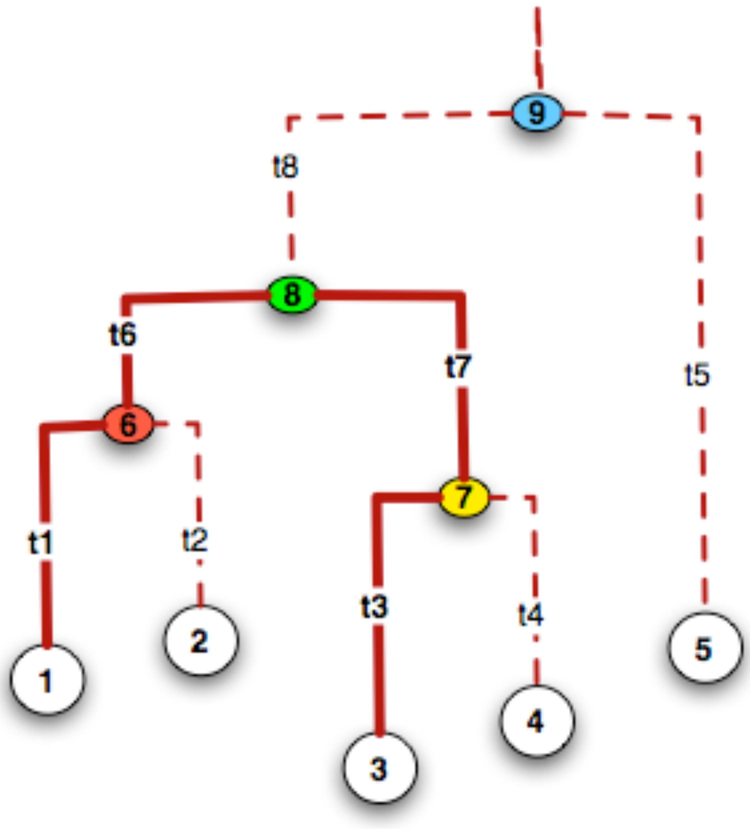
$$\sum_{a_8} P(a^1 | a^8, t_1 + t_6) P(a^3 | a^8, t_7 + t_3) q_{a^8} =$$

$$P(a^1 | a^3, t_1 + t_6 + t_3 + t_7) q_{a^3}$$

$$\underline{P(x_u^i, x_u^j | T, t_\bullet) = q_u^j P(x_u^i | x_u^j, t_{k1} + t_{k2} + ... + t_{kr})}$$

$$d_{ij}^{ML} = \arg\max_t \left(\prod_u q_u^j P(x_u^i | x_u^j, t)\right)$$

$$d_{ij}^{ML} \simeq t_{k1} + t_{k2} + ... + t_{kr}$$



Next week's reading

- Ch. 9 BSA : Preparing for RNA structure prediction
- Berezikov, Cuppen & Plasterk. Approaches to microRNA discovery. NATURE GENETICS SUPPLEMENT. S2 VOLUME 38.JUNE 2006