BMI/CS 776 Spring 2008
Homework #2

Prof. Colin Dewey

Due Thursday, February 14th, 2008 by 11:59pm

The goal of this assignment is to become more familiar with the statistics and algorithms for simulating and reconstructing nucleotide evolution. We will restrict our models to those involving only substitution. In later assignments, we may consider additional mutation events (e.g., indels).

To turn in your assignment, copy all relevant files to the directory:
`/u/medinfo/handin/bmi776/hw2/USERNAME`
where `USERNAME` is your account name for the BMI network. You must submit a file named `README` to this directory, which gives directions on how to compile (if necessary) and run your programs. For each question below, the `README` file should list the files relevant to that question (e.g., code, other files with written answers).

1. Write a program, `nucsim5`, that takes as input a single DNA sequence, and outputs five DNA sequences that are the result of simulating nucleotide evolution at each position independently, along the tree in Figure 1 with the input sequence as the root. The five output sequences should be those at the leaves of the tree, $\{X_1, X_2, X_3, X_4, X_5\}$, after simulation. At each position in the sequence, substitutions should be modeled using the Jukes-Cantor matrix, a flat root distribution ($\mathbf{\Pi} = \frac{1}{4}\mathbf{I}$), and with the branch lengths indicated in Figure 1. Note that the branch lengths are specified as the *expected number of mutations* at each site along the branches (warning: not the same as λt, you will need to do a simple transform to get λt).

   To be clear about how we are simulating on the tree: given the base at a position in the root sequence, we first simulate the characters of each of the root's children by drawing from the distribution given by the conditional probability table (the substitution matrix, parametrized by the branch length). Given the children characters, we proceed to simulate the grandchildren, and so on.

   The syntax for running the program should be:

   ```
   nucsim5 root.fasta leaves.fasta seed
   ```

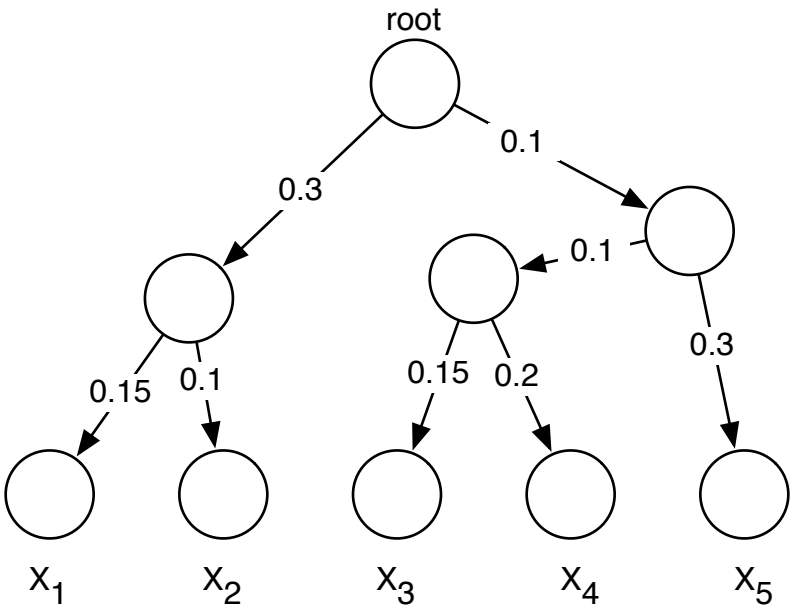

Figure 1: Phylogenetic tree relating the sequences to be simulated.

where `root.fasta` is the name of a file containing a single root sequence, `leaves.fasta` is the name of the output file, and `seed` is an integer for seeding the random number generator that you choose for simulation. Sequences should be read and written in FASTA format. The leaf sequences should be written to the output file with the leaf names (i.e., `x1`, `x2`, etc) as descriptions.

2. Write a program, `infermarg5`, that takes as input five DNA sequences, and outputs a single root ancestral sequence that is computed using the *marginal reconstruction* algorithm, with the tree model from problem 1. The syntax for running the program should be:

   ```
   infermarg5 root.fasta leaves.fasta
   ```

   where `root.fasta` is the output filename and `leaves.fasta` is the name of a file containing five sequences labeled `x1`, `x2`, ..., `x5`.

3. Write a program, `inferjoint5`, that is the same as `infermarg5`, but that uses the *joint reconstruction* algorithm instead.

4. Search NCBI's (`http://www.ncbi.nlm.nih.gov/`) nucleotide database for the DNA sequence `NC_001807`. Download the sequence in FASTA format (use the drop-down box labeled "Display" to select this format). What sequence is this?

Use `nucsim5` to generate five sequences evolved from this one. Run both `infermarg5` and `inferjoint5` on these simulated sequences to attempt to reconstruct the orginal sequence. How many positions did these algorithms correctly reconstruct? How many positions were reconstructed incorrectly by both methods? If the methods differ in performance, why do you think one was better than the other?