CSE 6[1]20: Project 3
Due: Sept. 27, 2013

Part 1: Newick Parsing

***Submission**:* All code will be added to the file *parseNewick.py* in your project 3 directory.

Purpose: You are writing a single function *parseNewick* that takes a Newick string and returns an appropriate PhyloTree object.

[1] By "we" I of course mean: chemists, biologists, and other people who are generally not me.

Part 2: Sequencing by Hybridization

The *sequencing* problem is determining the sequence of a DNA, RNA, or amino acid molecule. Solving it requires a combination of wet-lab and "dry-lab" (computational) techniques. Generally: through wet-lab work we can find the exact sequence of small chunks of the molecule (though this is frequently somewhat error prone), which we then need to *assemble* into an entire string through computational techniques.

In Sequencing by Hybridization (SBH), the chemists can give us a multi-set of all *l*-mers (strings of length *l*) occurring in our sequence – called the *Spectrum*. For example, if *s = "TGCAT"* and *l=3*, then *Spectrum(s,l) = {CAT,GCA,TGC}* – every length 3 substring of *s*. The goal of SBH is, given *l* and a multi-set *S*, to find a string *s* such that *Spectrum(s,l) = S*.

For this assignment, we will assume we are working with a *circular* genome – a characteristic of most bacterial genomes. Hence we can have substrings over then "ends". For example, if the genome *s = "ACGT"* is circular, than *Spectrum(s,3) = {ACG, CGT, GTA, TAC}*.

Formally:

- Input: A list *S*, representing an unordered multi-set of length *l* sequences.
- Output: A string *s*.
- Goal: *Spectral(s,l) = S* when *s* is taken as a circular genome.

Clarification: Note that while *S* is a list, it represents a multi-set – hence its order has no significance.

Hint: This is, in principle, very similar to the TSP. You can use a variation on the TSP branch-and-bound algorithm (from lecture) to solve it.

Comment: There is actually a polynomial time approach to this problem (see section 8.8 of your text). However, the point to this question is to focus on branch-and-bound techniques. So you are not responsible for learning or using polynomial-time approach. You may do so if you prefer – but choose that option only if you are sure you know how to implement the branch-and-bound solution.

Submission: Code is to be written in the **proj3/SBH.py** file. The **SBH** function is to be used as the main function, though any necessary helper functions may be added.

Grading:

- 90% of the points for this question will be assigned based on correctness.
- 10% of the points for this question will be assigned based on speed. Solutions must be fully correct to earn any points for speed.