CMSC423 Project 2
Handed out: 11/11/2008
Due: 12/9/2008
Project choice due by 11/18/2008

The second project in the class is your choice. For convenience I've listed four possible projects below, however you can come up with your own project. Furthermore, for this project you can work in a team of up to two people. You have until next Tuesday (11/18/2008) to email me both your project choice and the name of your partner.

If you choose to work as a team, the project will have to be more complex than if you work alone - I have outlined how in the description of the projects below.

Note that the description of the projects is intentionally vague - part of your assignment is to figure out what will make your project good.

Project deliverables:
1. Your source code (should run on the glue machines)
2. Documentation describing how to run your code.
3. Report (1-2 page) describing the results of applying your code to some public data. The report should also highlight potential improvements/extensions you might want to implement if you had more time available.

1. Gene finder
Implement a simple gene finder for bacteria, specifically find stop codons, identify in-frame starts, then evaluate the ORFs, based on codon preferences in the organism being analyzed. Your program must accept as input the sequence of an organism as well as a codon bias table (http://www.kazusa.or.jp/codon/). As output you should provide a list of genes in GFF format (http://www.sanger.ac.uk/Software/formats/GFF/)

If working in a team you must also implement an additional module that identifies long stretches of DNA without a valid ORF,then searches these regions against a protein database using the blastx program in order to identify genes potentially missed by the initial module.

2. Shotgun sequence overlapper
Write a program that performs the pairwise comparisons between the set of sequences provided as input to a shotgun sequence assembler and report which sequences overlap. This project will require you to use a combination of exact matching (e.g. k-mer hashing) and inexact matching (Smith-Waterman) techniques. Your program must accept as input a set of sequences in FASTA format and must produce the output in AMOS format (see http://amos.sourceforge.net - the RED and OVL records).

If working in a team you should also integrate your program in the minimus assembler (component of AMOS) and compare its performance with the original implementation.

3. Multiple aligner
Implement a simple multiple alignment program, extending the Smith-Waterman algorithm implemented in project 1 to allow implementing the progressive alignment approach. As input you must accept a set of protein sequences in FASTA format, then output their multiple alignment in ClustalW format (http://www.bioperl.org/wiki/ClustalW_multiple_alignment_format).

If working in a team your aligner must use a guide tree, and you need to evaluate the impact of the tree structure on the multiple alignment (e.g. implement UPGMA and neighbor-joining and compare the quality of the alignments).

Test data and reference alignments can be obtained from BaliBase (http://bips.u-strasbg.fr/fr/Products/Databases/BAliBASE/).

4. RNA folding
Write a program that computes the secondary structure of an RNA molecule. The input to your program will consist of an RNA sequence in FASTA format., and the output must be presented in parenthesized form:

```
>Sample RNA
AAAAAAAAAAAAAGGGGGGGUUUUUUUUUUUUUUCCCCCCCCCCCCCCCCCC
.............((((((.............))))))..........
```

Note: RNA sequences can be obtained from the NCBI: http://www.ncbi.nlm.nih.gov

If working in a team, your aligner must take into base-pairing energies as well as a stacking term (energy of stem depends on number of stacked bases in a way similar to affine gap penalties in Smith-Waterman). Energy values can be found at http://www.bioinfo.rpi.edu/zukerm/cgi-bin/efiles-3.0.cgi. Note: simplify the information on this website - just pick two sets of energies for each pair of bases - one energy if the bases are part of a stem, another if they are not.