Computer Science 549 – Computational Biology
Prof. Steven Skiena
Fall 2013

Semester Project
Proposal due: Thursday, October 24, 2013
Progress report due: Thursday, November 14, 2013
Project due: Thursday, December 5, 2013 (in class)
Informal Presentation: Friday, December 6, 2013 (my office)

September 26, 2013

The project will involve concentrated work in one research project related to computational biology. Below I list several possible projects. You may also choose your own topic if you can convince me that what you want to do is interesting.

My hope is that projects will involve interaction between life science and computational students. Thus there is a rule that every group *must* have at least one Computer Science student to prevent the non-CS students from clumping up together.

While I anticipate that much of the work will be done as the deadline approaches, it is important to get started early enough to discover insurmountable roadblocks in data acquisition or problem definition before it is too late. The project proposal and progress reports have been instituted to ensure people get serious well in advance of the final deadline.

Each group is responsible to turn in 3-5 page project proposals/literature search and progress reports as of the dates above. I will award roughly 50% of the grade for each project on the strength of the preliminary reports. This is to encourage starting early, and to make sure that you and I both know what you are to do before it is too late to avoid trouble. Each group will have to turn in a final written report/WWW site and have a quick meeting with during finals week – during which I will ask if all members participated equally.

My hope is that one or more of these projects will lead to published work. This has been the case most times I have taught such a course. The projects I believe are best have been starred (*). Fresh projects for this year are marked (F). Those marked PhD are recommended for groups of PhD students. I particularly recommend them to students thinking about doing research under me.

1. *Oxford Nanopore / PacBio Assembly* (* MS/PhD) - new sequencing instruments from Oxford Nanopre and Pacific Biosciences can sequence much longer fragments of DNA than any other sequencing technology over 2000bp compared to 100-500bp), but at a much higher error rate (typically 15% error). The long read length makes the instruments very attractive for de novo assembly of complex genomes, but the high error rate prevents traditional approaches from being used. Design and implement an algorithm for assembling these reads de novo, such as using spaced seeds for finding overlaps or an error correction pipeline that can be applied directly to the reads. Requires strong algorithm and programming skills.

   There is 30x to 100x coverage of long reads of a few strains of E. coli publicly available from PacBio: http://www.pacbiodevnet.com/Share/Datasets/E-coli-Outbreak Assembling E.coli from just long reads should be challenging but possible. Check out http://schatzlab.cshl.edu/teaching/2013/pacbioasm.shtml for resources – we are most concerned with developing a good overlapper for these reads. (joint with Michael Schatz, mschatz@cshl.edu and Skiena)

2. *Queuing Service for High-Throughput Sequencing* (*F) – High-throughput sequencing machines are expensive to run (say $30,000 per run) but have so much capacity that each run can be shared between several experiments to amortize the cost per experiment. Thus machine holders wait until the machine is full, and may offer discounts for the final experiments, while experimenters are looking for the fastest, cheapest machines.

   This would suggest the need for a website matching machines with people who need sequencing. This project asks you to build such a thing, including making the contacts to get suppliers interested in participating and figuring how to promote this service among labs in need. The best contact here is probably Justin.Gardin@gmail.com. (This is a good project for MS CS people with web development experience)

3. *Phylogenic Trees of Languages* (*F) – Produce a phylogenic tree of all Wikipedia languages from a mix of sources. Start by doing (a) a literature review on language phylogenies, and (b) analysis of the words extracted from Wikipedia/Wiktionary. Try both feature-based approaches (which languages contain which words) and distance-based approaches (score how similar each pair of languages are).

4. *Most Interesting Unsequenced Genomes* (F) – Do a study to rank the most interesting species whose genomes should be sequenced. Interest in a genome grows with the phylogenic distance to the nearest sequenced species, plus perhaps with some consideration of which species are discussed in the scientific and popular literature and how much partial sequence data already exists in databases.

5. *Group testing on Continuous Variables* (*F) – We have developed a group-testing based signal detection proceedure will identify one lethal defect in 16 regions with four designs – see Skiena "Redesigning Viral Genomes". Further, we have developed more sophisticated designs with robustness properties: check out our paper "Synthetic Sequence Design for Signal Location Search", RECOMB 2012.

   A generalized version of our signal search problem seeks to estimate the relative fitness of each of n regions of a gene, given multiple alleles for each region. This problem also arises in protein design, where the desired proteins have multiple do- mains, with multiple different choices for each domain. For example, Rawat, et. al. ("Conjugated Fatty Acid Synthesis", J. Biological Chemistry) optimizes the design of a conjugase enzyme with six domains and two choices per domain by building/assaying only eight of the possible $26 = 64$ constructs. Cost issues make it impossible to build and assay all designs in any substantial search space – the exact same situation motivating our signal search procedure.

   Propose families of designs and evaluate them (analytically and via simulation), developing appropriate algorithms (**likely based on linear programming**) to interpret the resulting under-constrained systems of linear equations. (from John Shanklin at BNL). *Need:* Algorithmically-intensive CS project.

6. *Detecting Signals By Homology* (*F) – Firth, et. al ("Stimulation of Stop Codon Readthrough", Nucleic Acids Research 2011) developed a homology-based method to identify signals in coding sequences. He reasons that such signals should be conserved in closely-related species under the proper alignment, where he aligns the amino acid sequences and then looks for residues with unusually high conservation of codons at each position. His results are impressive, but not always sensitive enough. Two directions remain open to improving such signal search procedures:

   - *Selecting homologous candidates* – The most difficult aspect of applying Firth's methodology in practice is identifying the right set of database sequence candidates to align. Sequence variants which are too closely related to the target are so highly conserved as to be valueless, while too distant homologies are hard to properly align and may well mask the signal. Give an automated method to suggest appropriate signals analysis of a BLAST query to screen a large set of possible candidates and select the most informative sequences.
   - *Improved statistical methodology* – Firth's statistical methodology is sophisticated, but he bases his analysis only on the amino acid residues which align identically to the target at each position. This often means discarding the majority of the candidate sequences from weighing in at any

particular position. We believe this additional information can improve our analysis, particularly when coupled with phylogenic tree data.

7. *Gene Design for Self-Assembly* (*F) – Self-assembly requires designing sequence pairs whose ends hybridize together. The biggest design challenge for self-assembly is to ensure that the pieces fit together in only one way, i.e. that there be no unintended cross-hybridization among the oligos. A class of synthetic sequence design tools, including DNAWorks, GeMS, and GeneDesign focus on optimizing designs for manufacturability (i.e. selecting oligos without local secondary structures and end repeats). The 200bp oligo lengths of contemporary Agilent synthesis are luxuriously long relative to shorter oligos used in traditional assembly. These means we can design somewhat shorter oligos with variable lengths (say an average length of 180 plus/minus 20bp) so we can stagger the entry points to these repeats in an effort to ensure they assembly in the right way. Develop an algorithm to do so.

8. *Visualizing Higher-Dimensional Word Embeddings* (*F) – Word embeddings for a given language (*https://sites.google.com/site/rmyeid/projects/polyglot*) are high (50-300) dimensional vectors for each of the 100K+ words in the vocabulary of a language. But they are hard to understand. Visualize them by (1) permuting the dimensions so as to maximize the correlation with neighboring dimensions, and (2) permuting the words according to the agglomerative clustering algorithm / TSP to maximize coherence. Part (2) is done by popular microarray clustering programs – produce the right programs/visualizations of embedding data.

9. *Data Mining/Analysis Projects* (*) – Experiment with a data mining/analysis technology on an interesting problem / data set. This project is good **IFF you have a good application and idea of where you will get data from**. I want *new* data sets and *new* problems, and the task cannot be from something in another course.

   (a) *Building HMMs* – Build a general-purpose Hidden-Markov Model implementation, supporting one or more learning algorithms, general topologies, probability computations, etc.

   (b) *Using SVMs* – Use support vector machines (SVMs) to solve some classification problem and report the results.

   (c) *Using Baysian Networks* – Use Baysian networks to model data from an entity, citation, or other relevant network.

   (d) *Phylogenies of Texts/Disciplines* – Use phylogenic tree algorithms to reconstruct history of (say) geopolitical splits, language/cultural groups, or partitioning of academic fields..

10. *Properly Arranging Words* (*F) – Given a set of words in arbitrary order, can you find the most likely order for them to make a sentence? Use ngrams or word embeddings to make sense of this, with perhaps using a grammer.

11. *Theoretical Algorithm Problems (PhD only)* – Each of these requires at least one algorithmically strong CS-type, and a keen sense of when to give up and move to something else.

    (a) *The String Chomping problem* (*F) – As discussed in the reading group, study the complexity of string rewriting problems for a specific pattern (or group of patterns) going to null:

    - What is the complexity of finding the minimum possible remaining sequence under a given chomping pattern?
    - What is the complexity of testing whether there is only one fixed point, i.e. one final maximially chomped string, or can there be many such strings?

    These will have to be written up clearly enough and cleanly enough for potential publication.

    (b) *Shortest Subset Subsequence* – As discussed in algorithm reading group, what is the string on an alpabet of size n such that every one of the 2^n subsets is represented by a substring of minimum size, ie. equal to the cardinality. (Probably Difficult)

(c) *Finding the Most Frequent Subsequence* – Given a string S, which string S' occurs most often as a scattered subsequence of S? Note that the number of candidates and possible occurences are exponential. Can you prove this is NP-complete, and maybe give an approximation algorithm? Dynamic programming can count the number of occurences of a candidate S'. (Probably Difficult)

12. *Software Survey Papers* – Write a 10-15 page in-depth survey paper on the state of software available for one of the following topics of interest, or pick your own topic:

    - Gene design to optimize expression
    - DECODE project/results
    - Haplotyping
    - Synthetic biology
    - RNA interference.
    - Genetic linkage analysis.
    - Computational challenges in Epigenomics
    - Small RNAs, microRNAs and other noncoding RNAs

    In all cases, discuss the different programs available for the problem, and how they compare by performance, speed, input requirements, and algorithms. Your paper should be based on some actual experience with the programs.

    All survey papers must appropriately cite all sources, and be written in *your own words.* Any student caught plagerizing from Web or published sources will receive a Q grade and be subject to academic dishonesty proceedings. Don't be stupid – I know how to use Google, too. *Need:* each survey paper should be done individually by either a CS-type or a biologist

Life Science Projects

Certain projects have been contributed by faculty from life science departments affiliated with Stony Brook. Many are looking for student to do larger projects with, so this is an opportunity to make a good impression. Be polite and respectful of their time. I encourage you to discuss the project with them – however, contact me first by email with your intention so I can make sure that they are not flooded with responses.

1. *Elucidating the cause of immune system cancers* – B-cells are a vital part of the immune system whose principal function is to generate antibodies. During an infection, antibodies are created by means of an evolutionary process during which the antibody genes are mutated. Only those B-cells that generate mutations leading to better antibodies survive. How the mutations themselves are generated is a subject of intense research, but we now understand quite a few details. During the process of transcribing RNA in any gene, the RNA polymerase molecule pauses and undergoes a transition from an early initiation phase to the later elongation phase. The evidence suggests that many antibody gene mutations occur during this transition. Although B-cells mostly manage to mutate only the antibody genes, mutations can sometimes hit other genes and cause cancer. We would like to know whether these off-target mutations also occur during the initiation-to-elongation transition. Existing data from the ENCODE database can be used to identify the regions where this transition occurs in many genes. These results can then be compared to the positions of known B-cell cancer mutations to understand whether the off-target mutations are also occurring during the initiation-to-elongation transition. (from Tom MacCarthy, thomas.maccarthy@stonybrook.edu)

2. *Find the 5' and 3' ends of bacterial 16S RNAs* (from Bruce Futcher)– Databases with hundreds of thousands of DNA precursors of the 16S RNAs exist, but these are of DNA sequences, not modified RNAs. So: (1) Do careful alignments of the end regions as per the 16S phylogenic trees for insight and (2) Look for Shine Delgarno sequences as enriched upstream regions.