CSE 308/408 Bioinformatics: Issues & Algorithms

Location: Mohler Lab 110, Mondays, Wednesdays and Fridays, 1:10pm – 2:00 pm
Alternate Location: Packard Lab 122, "Sunlab," for experience on enterprise computing systems

Professor: Brian Y. Chen, Department of Computer Science and Engineering, Lehigh University

1. Course Description

A sequence of nucleotides forms a blueprint for the astoundingly complex systems of interacting molecules that are the basis for all life. This information, stored on an atomic scale in all living things, has remained inaccessible until the emergence of genomics, a field of technologies for determining nucleotide sequences, and largely incomprehensible without the development of bioinformatics, a computational field for assembling, comparing, and analyzing nucleotide sequences. Together, these fields sparked an explosion in biological and biomedical research frequently called the "biotech revolution". This course explores the biological principles and the computational theories that fuel this ongoing revolution, with a focus on genome assembly, annotation, and evolution.

2. Textbooks and Course Materials (Optional)

J. Pevsner, Bioinformatics and Functional Genomics, Wiley-Blackwell, 2nd ed.
N.C. Jones and P. Pevzner, An introduction to Bioinformatics Algorithms, MIT Press, 1st ed.
Textbooks are intended to supplement lectures, which are the primary course material.
CourseSite: Online resources will link lectures, assignments, and supplemental readings.

3. Course Organization

CSE 308/408 is purposefully designed to be simultaneously accessible to biological students with no computing background and to computational students with no biology background. **The course can be completed without programming.** Simultaneously, the course crosstrains students in their non-specialty discipline. This didactic purpose is achieved through three major mechanisms:

- Three core projects, which each have an "applications" variation and an "implementation" variation. Applications projects require a deep understanding of biology and the usage of enterprise computing systems (e.g. Unix), but no programming. Implementation projects require a sophisticated algorithmic background, and depend less on an existing background in biology.
- All students complete their own project, but are paired with another student that is anticipated (though not required) to complete a project in the other discipline. Topical discussion and collaboration is strongly encouraged, with extra credit for documented instances of productive collaboration. Considerable additional extra credit is possible for groups that are able to nontrivially integrate their project software and data.
- Projects are paired with reports that probe the student's crossdisciplinary understanding. Applications project reports that require a conceptual description of how a major algorithms in the field work. Implementation project reports require the conceptual application biological principles in hypothetical experimentation. Reports strongly motivate crossdisciplinary communications, which are further encouraged with extra credit (see above).

4. Lecture Topics

Quarter 1: Jumping in

Lecture 01: The Pervasive Nature of Computation in Science
Lecture 02: Introduction to Molecular Biology
Lecture 03: How Molecular Biology Translates to Computer Science
Lecture 04: Introduction to Unix
Lecture 05: Editting, Automation, and Remote Access

Quarter 2: Genome Sequencing

Lecture 06: Pyrosequencing
Lecture 07: Genome Sequence Assembly
Lecture 08: Hands on with SSAKE
Lecture 09: Sequence Assembly Algorithms
Lecture 10: Hashing in Sequence Assembly
Lecture 11: Solexa Sequencing
Lecture 12: Exploiting Paired End Reads
Lecture 13: Path Compression in Genome Assembly
Lecture 14: Biomedical Applications of DNA Arrays

Quarter 3: Genome Annotation

Lecture 15: Genome Annotation
Lecture 16: Finding Genes with Genemark and Glimmer
Lecture 17: Nucleotide Sequence Alignment Part 1
Lecture 18: Markov Models for Genome Annotation
Lecture 19: Nucleotide Sequence Alignment Part 2
Lecture 20: Dynamic Programming with Affine Gap Penalties
Lecture 21: Backtracking on Affine Gap Tables
Lecture 22: Applications of Sequence Alignment and Gene Prediction

Quarter 4: Genome Annotation

Lecture 23: Genome Evolution
Lecture 24: Phylip and Phylogenies
Lecture 25: Building Phylogenetic Trees 1
Lecture 26: Building Phylogenetic Trees 2
Lecture 27: Phylogeny and Molecular Biology
Lecture 28: Finding Active Sites through Phylogenetic Analysis

5. Sample Projects

Project 1: Genome Annotation

- Applications Project:

  You are trapped on a derelict space ship with a broken genome sequencer, a copy of NCBI's viral genome database on a flash drive, and famed genome scientist J. Craig Venter, who has fallen sick of a mysterious virus. Beginning with a mixture human and viral reads, sequence only the viral genome and determine the most similar Earth virus.

- Implementation Project:
  Implement a genome assembler based on the principle of Sequencing by Hybridization. A dynamic hashing structure, a Fasta parser, and real genome sequencing software, are provided for your development effort.

Project 2: Genome Annotation

- Applications Project:
  You have sequenced the unknown virus afflicting Dr. Venter and the crew, but you have yet to find what genes it has – a crucial step towards trying to find any possible treatment. Using Genemark and Glimmer, annotate the viral genome
- Implementation Project:
  Implement nucleotide sequence alignment software with affine gap scoring. A Needleman-Wunsch implementation using affine gap scoring and several examples and provided for your development effort.

Project 1: Genome Evolution

- Applications Project:
  Reconstruct a balanced, unbiased phylogeny of a given protein and identify the amino acids that are the most probable members of an active site
- Implementation Project:
  Implement UPGMA and Neighbor-joining algorithms for reconstructing the phylogeny of a family of proteins from a distance matrix. Several examples and PHYLIP, a popular and effective tool for generating UPGMA and Neighbor-joining tree, is provided for your development effort.

6. Outcomes

By completing this course, students will:

1. Understand the design and purpose of several major computational technologies in the field of Bioinformatics.
2. Be aware of how biological, algorithmic, and statistical concepts can be integrated to draw meaningful conclusions from multi-faceted biological data.
3. Have experience or a conceptual understanding of the implementation challenges relating to these major technologies.
4. Have experience in technical communication with collaborators with technical expertise outside of their own field.

This course supports program missions to educate students that will:

1. Apply their education in computer science to the analysis and solution of scientific, business, and industrial problems.
2. Function effectively in a collaborative team and effectively communicate with members of the team.

7. Assessment

Projects: CSE308: 3 projects, 25% each. CSE408: 3 projects, 20% each.
Students will work in pre-assigned groups of two, consisting of one BioE student and one CSE student. Imbalances in available class backgrounds will be resolved through individual or three-person groups. Each group member will complete their own applications project or

implementation project (50% of project), as well as a report (50% of project). Projects can be integrated for up to 50% extra credit on each project. In each project, students in the same group receive the same score: the average of their scores. Dysfunctional groups will be separated. No group will repeat a pairing of the same students in different projects.

Methods projects require C/C++/Java programming. Applications projects require only scripting and the usage of a Unix/Linux environment. Computational students are expected to assist their biological partner with scripting or computer usage questions. Students with a biological background are expected to assist their computational partner with any biological questions. These topics will also be covered in the course introduction.

Up to 25% extra credit is possible via detailed documentation of collaborative efforts between students in the same group and in other groups. No more than 50% total extra credit is possible in any single project, even though 50% extra credit is possible via interdisciplinary project integration (described above).

Projects will be submitted via email by 12:01 am on the assigned due date with an 8 hour grace period. Due to generous extra credit rules, projects more than 8 hours late will not be accepted, and will receive zero credit.

Extra Credit for projects can only compensate for credit lost on the same or other projects, and thus cannot compensate for non-participation or not completing the CSE408 review paper.

Participation: CSE 308: 25%. CSE408: 20%
Credit for participation is evaluated in proportion to the number of lectures in which each student asks at least one question. For example, if a student asks at least one question in 80% of lectures, that student will receive 20% out of a possible 25% for this portion of the assessment. Obvious, trivial, or repetitive questions will not count.

CSE408 Review Paper: CSE308: 0%. CSE408: 20%
In addition to completing the three projects students enrolled in CSE408 will write a review paper outlining the current state of the art in one of three major fields discussed in this course: Genome assembly, Genome Annotation, Genome Evolution. The report is expected to be 10 pages, singled spaced with a 12 pt font and 1 inch margins, including bibliography.

8. Accommodations and Academic Integrity

Accommodations for Students with Disabilities

If you have a disability for which you are or may be requesting accommodations, please contact both your instructor and the Office of Academic Support Services, University Center C212 (610-758-4152) as early as possible in the semester. You must have documentation from the Academic Support Services office before accommodations can be granted.

Academic Integrity

The work you submit in CSE 308-408 must be entirely a product of your group. While you are encouraged to discuss basic concepts and strategies with friends and classmates, the copying or

sharing of solutions for projects is never acceptable. Such cases will be referred to the University Committee on Discipline and, if found guilty, you may be given the failing grade WF in the course. You should keep in mind that computer programs exhibit an individual's “style” just as much as other forms of authorship. Changing variable names, editing comments, reformatting or refactoring code, or making other trivial updates in an attempt to hide plagiarism is rarely effective. If you have questions about this policy at any point throughout the semester, ask. It is far better to be safe than sorry when your academic career may be on the line.