Foundations of Computing II
Course Description
Draft of May 26, 2009

Structural place in the curriculum:

- 4 credits (3 weekly lectures, 1 weekly section)
- Pre-requisites:
  - Foundations I is a pre-req
  - Programming Abstractions is a pre-req or co-req (see course overview for an explanation)
- Courses with Foundations of Computing II as a pre-requisite
  - 400-level theory courses (algorithms, complexity)
  - 400-level courses requiring probability or statistics (AI, data mining, vision, others)
- Taken by: All majors (CS and CompE)
- Note: The course would meet the ABET requirement of, “knowledge of probability and statistics, including applications appropriate to the program name and objectives.”
- Catalog description: To be determined

Course Overview / Goals:

Foundations II would have three parts conceptually, though the first two would be intermixed. The first would cover the fundamental concepts of counting, probability, and the properties of random variables. The second part would involve applications, the central limit theorem, and statistics. The third part would be on polynomial-time and NP-completeness. Each would be a significant segment of the course.

The main items needed from the Programming Abstractions course for Foundations II would be graphs and asymptotic runtime analysis which would be necessary for the complexity part of the course. Since that is in the last third of the course and those topics would be relatively early in the Programming Abstractions course a co-req would be okay.

Course goals include an appreciation and introductory understanding of (1) methods of counting and basic combinatorics, (2) the language of probability for expressing and analyzing randomness and uncertainty (3) properties of randomness and their application in designing and analyzing computational systems, (4) some basic methods of statistics and their use in a computer science & engineering context, (5) the distinction between tractable and (apparently) intractable computational problems and (6) methods and appropriate reasoning for showing tractability (e.g. dynamic programming) and intractability (reduction).

Evaluation and Homework:

The course would have roughly weekly homework assignments and exams. Most homework assignments would be in a “paper and pencil” style, but some exposure to appropriate software, such as R, would help explore applications of statistics.

Possible textbook: To be determined. The Rosen text, which is a likely text for Foundations of Computing I, has decent coverage for the first third of the course. Segments of other texts would be used to supplement this material for the remainder of the course.

Rough topic list:
(Numbers of lectures are approximate.)

- Counting and Binomial Coefficients: (3-4 lectures)
  - Sum and Product Rules
  - Product Tree Analysis
  - Inclusion-Exclusion
  - Pigeonhole Principle
  - Permutations and Combinations
  - Binomial Theorem
- Probability (11 lectures, total, with topics and applications interspersed and approximately 4 lectures on applications)
  - Topic: Basic Probability
  - Topic: Conditional Probability & Bayes Theorem
  - Topic: Random Variables, Expectation
  - Topic: Bernoulli Trials and Binomial (and Multinomial ?) Distribution
  - Topic: Linearity of Expectation
  - Topic: Variance
  - Topic: Tail bounds (Chebyshev and Chernoff)
  - Topic: Normal distribution and Central Limit Theorem
  - Application: Average Case Analysis
  - Application: Randomized Algorithms
  - Application: Hashing, Fingerprinting
  - Application: Load Balancing
  - Application: Entropy and Data Compression
- Statistics (6 lectures)
  - How to lie with statistics (1 lec)
  - Maximum Likelihood: binomial, normal (2 lec)
  - Hypothesis Testing: Likelihood ratio, Chi-squared test (?) Contingency tables (?)(2-3 lec)
  - Polling and sampling (?) (1 lec)
- Polynomial-time, Reductions, and NP-completeness: (8-9 lectures)
  - Poly-time algorithms + examples (4 lec); Especially non-obvious polynomial-time algorithms e.g. via Dynamic Programming.
  - Search problems vs decision problems (1 lec)
  - NP-completeness, SAT (1-2 lec)
  - Reductions (2 lec)
  - Why it is useful, important to crypto, practice, etc. Overview of what we know (1 lec)

Correspondence to Old Curriculum:

This course has no direct analogue in our current curriculum.

- CSE 321 has about 2.5 weeks on counting, probability, and applications of average case analysis and Bayes' rule; this would be a subset of the material in the new course. 321 currently covers up to the end of linearity of expectation in the above syllabus.

- Relation to our current Statistics course requirements: CSE majors are currently required to take one of STAT 390/391 or both STAT 394/395.
  - STAT 390 is a "classical" introduction to probability and statistics without CSE applications. Between 2/3 and 3/4 of CSE majors take STAT 390.
  - STAT 391 is, in name, aimed at computer science, with some time spent on clustering and machine learning, but without other CSE applications discussed above. It is taught once per year to 40 students. Roughly 1/4 of current CSE majors take STAT 391.
  - A small number of students take STAT 394/395. This is purely a probability sequence without significant statistical content.

  Under the proposed revision, none of these courses would be required. However we would hope that STAT 391 would evolve to assume the additional background that students would have from Foundations of Computing II and become an even stronger course for our students that would be a (senior) elective and some of our 400-level courses may require as a prerequisite.

- CSE 322 has no overlap with Foundations of Computing II. See Foundations of Computing I.

- CSE 326 in some offerings has had about 1.5 lectures that informally discuss NP-completeness at the end of the course.

- Currently, both CSE 421 and CSE 431 spend around 2-2.5 weeks on NP-completeness. Since neither course is a prerequisite for the other, both courses cover a lot of the same material on reductions and cover the same examples. CSE 431 additionally proves the NP-completeness of SAT from first principles. Having material in Foundations of Computing II would eliminate this duplication. The coverage in Foundations of Computing II is planned to be similar in spirit to its current coverage in CSE 421.

  After the revision, CSE 421 would be able to cover additional algorithmic methods, including coverage of linear programming for which there currently is no time and randomized algorithms which will be better enabled by the additional material in Foundations of Computing II.

  After the revision, CSE 431 would still cover the proof of the NP-completeness of SAT but would be able to rely on the understanding and reduction examples given in Foundations of Computing II. However, there would be significant material from its current prerequisite CSE 322 that would be missing that could take up this slack (NFAs and a proof of Kleene's theorem for example). Alternatively, the course might be able to cover more advanced topics.