Randomness and Computation
or, “Randomized Algorithms”

Mary Cryan

School of Informatics
University of Edinburgh

Randomness and Computation

- Interested in what we can compute (exactly, approximately) when we have the option of “tossing coins” in our computation.
- Of course, when introduce some randomness, we no longer have a deterministic algorithm. An algorithm which exploits random choices will either show variation in the answer computed or in the time-taken to return an answer. Or both!
- Though we will have variation in running-times and/or the answer returned, we will always aim to calculate the expected running-times, expected value returned. Or possibly we will prove bounds on running-times and/or values returned.

Syllabus

- Introduction Las Vegas and Monte Carlo algorithms (Simple Examples: checking identities, fingerprinting)
- Moments, Deviations and Tail Inequalities (Balls and Bins, Coupon Collecting, stable marriage, routing)
- Randomization in Sequential Computation (Data Structures, Graph Algorithms)
- Randomization in Parallel and Distributed Computation (algebraic techniques, matching, sorting, independent sets)
- Randomization in Online Computation (online model, adversary models, paging, k-server)
- The Probabilistic Method (threshold phenomena in random graphs, Lovasz Local Lemma)
- Random Walks and Markov Chains (hitting and cover times, Markov chain Monte Carlo, mixing times)

Textbook (essential for the course)

Course Webpage

I will provide slides for each lecture (and notes sometimes if appropriate), but the collection of slides will not be sufficient to learn the material well enough.

Slides will be made available on the course webpage: http://www.inf.ed.ac.uk/teaching/courses/rc/

However, I will also be setting us up as a class on "nb", and storing the slides/notes there. "nb" facilitates class-wide discussions about the material, by allowing you to highlight a patch of the document and start a discussion there.

You will need the book too!

Pre-requisites

There are not really any formal pre-requisites for this class, as students are coming from both 4th year and the MSc year.

Strong Maths is required, especially Discrete Maths and confidence in proving things.

I expect you to have covered an “Algorithms class” in the past, and to have have done well in it (can waive that if your Maths is very strong).

If you're not sure, come and speak to me.

Math you should know

You should know:

▶ The definitions of the main categories of asymptotic operators $O(\cdot), \Omega(\cdot), \Theta(\cdot)$, and how to reason about them.

▶ How to multiply matrices or polynomials, also basic linear algebra.

▶ Some probability theory, definition of expectation (1st moment) and variance (related to 2nd moment), linearity of expectation, simple probabilistic distributions and how they behave.

▶ Some graph theory.

▶ what it means to prove a theorem (induction, proof by contradiction, etc . . . ) and to be confident in your ability to do this.

Your own work (formative assessment)

▶ 2 or 3 tutorial sheets.

I will arrange a couple of lectures as “tutorial style” where we will discuss the answers (probably week 7 and week 11/12).

▶ Office Hours.

TBA.

Office hours give you an opportunity to ask questions about material and tutorial questions and to get feedback on your own work.

▶ Coursework 1 (due Thursday of week 5)

The first coursework for RC will be read and commented-on by myself/my-TA; however it will not be “for credit”. It is to give you experience solving problems and doing small proofs.
Coursework (summative assessment)

We have 2 Assessed Courseworks (problem-solving and proofs), and both will be marked to give you feedback. Coursework 1 is “just for feedback”, and Coursework 2 will be worth 30% of the course mark. Details are:

- Coursework 1. “Feedback-only”
  - OUT Mon, 30th Jan (Mon week 3)
  - DUE 4pm Thurs, 16th Feb (Thurs week 5)
  - FEEDBACK returned by Thurs, 2nd March (Thurs week 6)

- Coursework 2. “Worth 30%”
  - OUT Fri, 3rd March (Fri week 6)
  - DUE 4pm Tues, 21st March (Tues week 9)
  - FEEDBACK returned by Tues, 4th April (Tues week 11)

Feedback given will include marks to individual sub-parts of questions, comments on scripts to explain why marks were lost, plus a description of common errors.

Verifying polynomial identities

Suppose we are given two polynomials \( F(x) \) and \( G(x) \), where \( F(x) \) is expressed as a product of \( d \) “monomials” and \( G(x) \) is given as an expansion of \( x^i \) terms, with degree at most \( d \).

How much time does it take to verify whether \( F(x) \equiv G(x) \)?

Simple “multiply out” algorithm on \( F(x) \) (uses no randomness) gives the answer in \( \Theta(d^2) \) time. Other (deterministic) algorithm uses FFT to “multiply out” in \( \Theta(d \cdot \lg^2(d)) \).

A “monomial” is a term of the form \((x - a)\), for some value \( a \).

We will use randomness to test equivalence (without multiplying out \( F(x) \)).

Reading Assignment

Start reading Chapter 1 of “Probability and Computing” in preparation for lecture 2.