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)


- Blackwell's on South Bridge has a number of copies of the 2nd ed, guaranteed to match Amazon's price.
- You are welcome to work with edition 1 if you can find a cheaper copy.
I will provide slides for each lecture (and notes sometimes if appropriate), and upload them to the course webpage:
http://www.inf.ed.ac.uk/teaching/courses/rc/

Recordings of lectures (slides and voice) will be on Learn.

However, I will also set us up as a class on "nb", and store 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

Please take the “Self test” on the course webpage to assess whether you are prepared for this course.

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 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)

- 4-5 tutorial sheets.
  We will have 5 tutorials sprinkled through semester, weeks 4, 6, 7, 8, 10. Slots will be Tuesday 12-1 and (if needed) Wednesday 12-1.

- “Office” (bench) hours.
  Come and ask me questions from 10:40-11 Tuesday, or 12-12:25 Friday (on the benches near our lecture room).

- Coursework 1 (due Monday 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 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 20% of the course mark. Details are:

▶ Coursework 1. “Feedback-only”
  ▶ OUT Fri, 26th Jan (Fri week 2)
  ▶ DUE 4pm Mon, 12th Feb (Mon week 5)
  ▶ FEEDBACK by Mon, 5th March (Mon week 7)

▶ Coursework 2. “Worth 20%”
  ▶ OUT Tues, 27th Feb (Tues week 6)
  ▶ DUE 4pm Thurs, 15th March (Thurs week 8)
  ▶ FEEDBACK by Thurs, 29th March (Thurs week 10)

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 $F(x)$).
Verifying polynomial identities using random sampling

- We will choose a value for $x$ uniformly at random from the set of integers \( \{1, \ldots, 100d\} \).

- Then we will calculate $F(x)$ for this value, taking $\Theta(d)$ time ($\Theta(d)$ shown on board).

- Then we will calculate $G(x)$ for this value, taking $\Theta(d)$ time (different reason for $\Theta(d)$, shown on board).

- And then compare the two numbers ... answering “yes” if they are the same, “no” otherwise.

We can show:

- If $F(x)$ really is the same polynomial as $G(x)$, the random process always says “yes”.

- If $F(x)$ is different from $G(x)$, then it says “no” with probability greater than or equal to 99/100.
Start reading Chapter 1 of “Probability and Computing” in preparation for lecture 2.
The goal of computational complexity is to classify computational problems according to their intrinsic difficulty or “complexity”. It complements algorithms as we will be aiming to provide “lower bounds” to various problems. Among other things, we will talk about why the P vs. NP problem, one of the seven Millennium Prize Problems, is important, intriguing, and difficult!