Sasha Rubin

Curriculum contribution *June 2017*

Approach to computer-science curriculum

I believe that computer-science curricula should provide students with a rigorous foundation in computational thinking and in the reasoning required to design and develop computational systems.

Computational thinking is rooted in mathematical logic. Thus, even if there are not many dedicated logic courses, I believe that one can and should inject computational aspects of logic into existing topics, e.g., Boolean logic and satisfiability in courses on discrete mathematics, unique-readability of logical formulas into courses on parsers, first-order logic in courses on databases, etc.

Just as important as learning foundations, is developing creativity in students that will be useful in real-world scenarios. To this end, I like to motivate and illustrate with examples from robot and mobile-agent control, as well as to provide concrete algorithms, the simplest of which can be simulated by hand.

Contribution to the common curriculum

The only course in the common curriculum that I am qualified to teach is “Quantitative Reasoning”. Although I have not taught such a course before, I have the following initial thoughts about useful topics that can be illustrated without heavy mathematics:

- *Deduction*, illustrated using propositional logic to test the consistency of a set of compound propositions, as in logic puzzles.
- *Induction*, illustrated by playing Abbott’s inductive card game “Eleusis”.
- *Correlation does not imply Causation*, and other useful lessons from Huff’s book “How to lie with statistics”.
- *Randomisation*, illustrated by Fisher’s “Lady tasting tea” experiment.
- *Regression to the mean* is a potent source of deception, as discussed in David Colquhoun’s blog “DC’s improbable science”.
- *Guestimation*, or Fermi problems.
- *Estimating risk* using the micromort unit, defined as a 10^{-6} chance of death.

Contribution to the MCS major

My background in mathematics and computer science means that I can teach a variety of theoretical courses.

Of the courses currently listed on the Yale-NUS College website, I can immediately teach the following:

- Introduction to Computer Science
- Theoretical Computer Science
- Proof

- Linear Algebra
- Advanced Algorithms and Data Structures
- Discrete Mathematics
- Calculus

Courses that I would enjoy teaching, but that would require me a little more time to prepare include:
- Fundamentals Of Programming
- Applied Calculus
- One real variable
- Group Theory
- Topology

I could offer the following foundational course:
- **Logic for Computer Scientists**. Logic has been called the *calculus of computer science*, in analogy with the (differential and integral) calculi that are fundamental in engineering. Indeed, logic appears as part of the fundamental abstraction of computing machines (Boolean algebra on zeros and ones), rigorous software design (verification and synthesis), in programming languages (semantics), and algorithms (computability and complexity theory). This course will provide students with the fundamental ideas from logic that are relevant to a wide swath of computer science and AI, i.e., syntax, semantics, and deductive systems of propositional and predicate logic. Time permitting, we will also discuss temporal logics and non-monotonic logics.

I could offer the following advanced courses:
- **Multi-agent systems**. A central theme in artificial intelligence (AI) revolves around the concept of an agent, which is any entity that can interact with other agents and/or the environment using sensors and actuators. In many cases, one is interested in "rational agents" (which can include humans, robots, software agents), which try to achieve a specific goal, and whose actions are guided by this desire. A system that consists of a group of such interacting agents is called a multi-agent system (MAS). Examples include software agents on the Internet, driverless cars, humans or software playing multi-player card games, robots exploring new and dangerous environments, and biological systems such as cultures of bacteria and swarms of insects. In this course we will model MAS as games on graphs, a convenient mathematical model of many phenomena in mathematics and computer science that involve *interaction*. The course will provide students with the foundational mathematics and algorithmic tools for modelling and reasoning about MAS.
- **The automata-theoretic approach to verification and synthesis**. Formal methods provide algorithms for automatically determining whether a mathematical model of a system satisfies a specification (verification) or, alternatively, to automatically construct a system that satisfies a given specification (synthesis). This field, for which the founders and proponents won two different Turing awards, has been used successfully for complex systems, and many hardware and software companies use these methods in practice, e.g., verification of VLSI circuits, communication protocols, software device drivers, real-time embedded systems, and security algorithms. In 2000, Moshe Vardi and Pierre Wolper won the Gödel Prize (an annual prize for outstanding papers in the area of theoretical computer science) for a 1994 paper that launched a unifying paradigm, i.e., the automata-theoretic approach to verification and synthesis. The power of the approach is that it reduces the problems to classic problems in automata-theory, and so neatly separates the encoding from the combinatorics. The course will provide an introduction to the theory of

automata and demonstrate its applications to verification and synthesis.

Teaching to diverse students

I have some experience teaching mathematical and computational ways of thinking to students of all backgrounds: I taught an interactive class to non-mathematics majors at Cornell University titled “Algorithms and Termination”; I taught an undergraduate course on mathematical logic at the University of Cape Town to students from all sections of South African society; I briefly tutored mathematics in Cape Town to students, who spent much of their time lobbying the government for better school facilities, to prepare for their final high-school exams.