Lecture 2: Proof and Formal Systems

Dr John Levine

CS103 Machines, Languages and Computation
September 25th 2015

University of Strathclyde Science

The Rules of the Game

- CS103: Machines, Languages and Computation
- 20 credits, both semesters
- Two lecturers:
  - John Levine (Semester 1)
  - Anders Claesson (Semester 2)
- Assessed by 4 class tests (30%) and an exam (70%)
- Exemption from the exam is possible if you do well in the class tests

University of Strathclyde Science

The Rules of the Game

- Lectures are Mon 10 in RC 345, Fri 10 in RC 641
- Tutorials: Thu 10 JA 505, Thu 1 in TG 227
- Lectures run Weeks 1-11 (no lectures in Week 12)
- Class materials will appear on the class web page:

  http://www.cis.strath.ac.uk/~johnl/CS103
- Facebook page: MLAC 2015
- Email: John.Levine@strath.ac.uk
- Room LT1420, extension 4524

University of Strathclyde Science

Tutorial Allocation

- You attend **one** tutorial per week, starting **next week**
- The Thursday 1pm tutorial is for those students taking:
  - BSc Computer Science
- The Thursday 10am tutorial is for those students taking:
  - MEng Computer Science
  - BSc Mathematics and Computer Science
  - MEng Computer and Electronic Systems
  - BEng Computer and Electronic Systems
  - BSc Software Engineering
  - BSc Business Information Systems
  - Everyone else not already mentioned

University of Strathclyde Science

Weekly Exercises

- Every week, you do a classwork assignment consisting of reading and some pen-and-paper exercises.
- This should take about 1-3 hours and can be done on the train, in bed, between other lectures, etc.
- The first three classwork assignments are contained in a workbook which will be handed out today.
- You hand in the workbook to the departmental office every week (details at the end of this lecture).

University of Strathclyde Science

Class Tests

- The classwork is assessed by means of a class test: a mini-exam in the usual lecture slot
- There are 4 class tests for CS103: two this semester and two next semester
- The Semester 1 tests are on Friday 30th October and Friday 4th December.
- Feedback and results will be available 2 weeks after the test date.

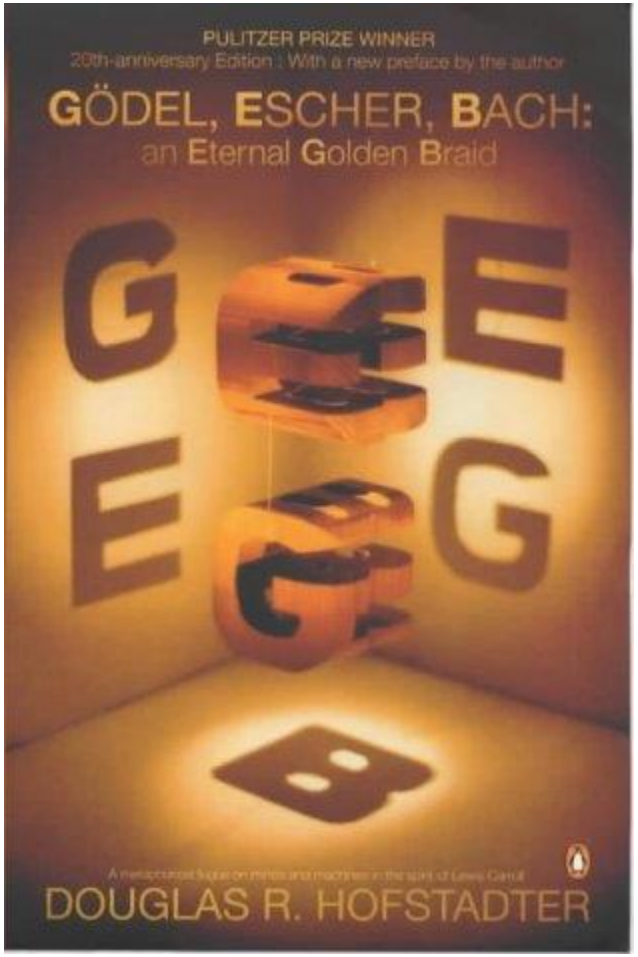
University of Strathclyde Science

Course Book

"Gödel, Escher, Bach: an Eternal Golden Braid"
by
Douglas R. Hofstadter

Publisher: Penguin Books Ltd
ISBN: 0140289208
Cost: £18.99 (amazon.co.uk)

You will need your own copy of this book as we will be reading from it every week.



All Computers are Equivalent

- In mathematical terms, all computers are equivalent (in the same way that all cars are equivalent).
- If you have a problem which can be solved on one computer, then it can be solved on *all* of them! This fundamental idea is called *The Church-Turing Thesis.*
- There are some problems which cannot be solved on *any* computer, no matter how powerful the computer is.
- The mathematical properties of *all* computers can be exactly modelled by very simple abstract devices.

University of Strathclyde Science

Non-Computable Functions

- The first 5 weeks of the course will build up towards the proof that non-computable functions exist:

  If P is the set of all computer programs and F is the set of all functions $f: n \rightarrow m$ such that n,m are members of the set of natural numbers N, then $|F| > |P|$ and therefore there are some functions in F for which no computer program can exist.

University of Strathclyde Science

Non-Computable Functions

- The first 5 weeks of the course will build up towards the proof that non-computable functions exist:

  If P is the set of all computer programs and F is the set of all functions $f: n \rightarrow m$ such that n,m are members of the set of natural numbers N, then $|F| > |P|$ and therefore there are some functions in F for which no computer program can exist.

University of Strathclyde Science

What is a Mathematical Proof?

- In mathematics, a proof is a logical argument showing that some statement (a *theorem*) is necessarily true
- A statement is an expression that is either true or false, such as “for every integer $n > 0$, the sum of the first n odd numbers is equal to n^2” or “all even numbers can be expressed as the sum of two prime numbers”
- The first of these can be proved – we will prove it later in the class
- The latter statement has not been proved – it is known as “Goldbach’s Conjecture”

University of Strathclyde Science

Infinite sets and proof

- The theorems we are trying to prove often have to hold true for infinite sets of numbers
- Example: prove that the sum of *any* two even numbers is also an even number
- The two numbers chosen can be any two members of the set of *all* of the even numbers (an infinite set)
- We deal with this by not naming specific numbers, but by using *letters* to stand for any two even numbers
- If our argument works just using those letters, we know it will work for any two specific even numbers

University of Strathclyde Science

A Simple Proof

Theorem: adding two even numbers always gives an even number.

- Proof: let the sum be $s = a + b$ where a and b are *any* two even numbers
- Let $p = a/2$ and $q = b/2$. Because a and b are even, p and q are both whole numbers (integers)
- The sum is now: $s = 2p + 2q$
- We can rewrite this as: $s = 2(p + q)$
- Because p and q are integers, $(p + q)$ is also an integer
- s is 2 times an integer, and so this means that s must be an even number. ■

University of Strathclyde Science

Types of Proof

- *Direct proof* is where the conclusion is established by logically combining the axioms, definitions and earlier theorems.
- *Proof by contradiction* is where it is shown that if some statement were false, a logical contradiction occurs, hence the statement must be true.
- *Proof by induction* is where a “base case” is proved, and an “induction rule” used to prove an (often infinite) series of other cases. Since the base case is true, the infinity of other cases must also be true.

University of Strathclyde Science

Formal Systems

- The proof we've just seen is a mixture of *formal* symbol manipulation (e.g. rewrite "$s = kx + ky$" as "$s = k(x + y)$") and more informal logical argument
- To be totally sure of our arguments, we can try to make *all* of the steps be formal symbol manipulation
- If we have symbol manipulation rules which can only produce true statements from true statements, our proofs are cast-iron guaranteed
- This could also allow computers to construct proofs!

University of Strathclyde Science

Gödel, Escher, Bach, Chapter 1

- In Chapter 1 of GEB ("The MU Puzzle") we are given the axiom MI and we have to make the theorem MU, using these four purely symbolic rules:

  I. $x\mathrm{I} \rightarrow x\mathrm{IU}$
  II. $\mathrm{M}x \rightarrow \mathrm{M}xx$
  III. $x\mathrm{III}y \rightarrow x\mathrm{U}y$
  IV. $x\mathrm{UU}y \rightarrow xy$

- Hofstadter asks: can you make MU?
- Levine asks: given some *arbitrary* string, can I write a computer program to determine whether or not it is a theorem?

University of Strathclyde Science

Homework

In the workbook, try these exercises (in this order):

- Assignment 2, parts (a) and (b)
- Assignment 1, parts (a) and (b)

Please hand in your workbooks to the department office by ***3pm on Tuesday 29th September***

Reminder of your tutorial for next week:

Thu 1pm in TG 227 for BSc Computer Science students

Thu 10am in JA 505 for everyone else