Historical Foundations

CS605 - Math & Theory of CS
T.J. Naughton
Maynooth University, Ireland

International Congress of Mathematicians, 1928

- Mathematician David Hilbert of Prussia (now Russia) asks the assembly if one can prove that mathematics is
  - complete; i.e. every mathematical statement can be either proved or disproved.
  - consistent; i.e. no contradictions, statements such as "0 = 1" cannot be proved from axioms.
  - decidable; i.e. is there a mechanical method of finding the truth of every assertion.

Vienna, 1931

- Kurt Gödel of Austria-Hungary (Czech Republic today) answered two of Hilbert's questions.

- He showed that every sufficiently powerful formal system is either consistent or incomplete.
  - If an axiomatic (rule-based) system is consistent, this consistency cannot be proved within itself (i.e. it is incomplete).

Incompleteness theorem, 1931

- In his proof, Gödel required some previous discoveries concerning problems with self-referential statements...

Incompleteness theorem, lead-up

- In 1901, the Welsh logician Bertrand Russell had discovered that *self-referential* terms such as ‘set of all sets’ could be contradictory (Russell’s paradox).
- In the eyes of many mathematicians (including Hilbert) it appeared that no proof could be trusted once it was discovered that the logic apparently underlying all of mathematics was contradictory.

Incompleteness theorem, lead-up

- Russell did manage to produce a solution, however, with his theory of *types*.
- Using the ‘vicious circle’ principle (of Henri Poincaré) he limited the use of self-application references to objects on the same level or *type*.
- But the self-referential loophole had been discovered, and was open to exploitation.

Incompleteness theorem, 1931

- Gödel was able to show that there were (undoubtedly true) statements about numbers which could not be proved from finite axioms by a finite number of rules.
- His proof rested on the idea that statements *about* numbers could be coded *as* numbers, and he constructed a self-referential statement that answered two of Hilbert’s questions.

Incompleteness theorem, 1931

- Both Russell’s and Hilbert's attempts at formalism were dealt a severe blow by Gödel's results, and ended a hundred years of theorist’s attempts to put the whole of mathematics on an axiomatic basis.
- Formal systems would have to be more comprehensive than previously envisaged.

Incompleteness theorem, 1931

- Gödel's results were of the most important in 20th century mathematics, showing that mathematics is not a finished object, as had been previously believed.
- The importance of this for computer scientists is that Gödel's results also implied that *a computer could never be programmed to answer all mathematical questions!*

Entscheidungsproblem, 1931

- Hilbert’s third question remained open,
  - decidability; is there a mechanical method that can be applied to any mathematical assertion which (at least in principle) will eventually tell whether that assertion is true or not?

  where *'provable'* could be substituted for *'true'*.
- At the time, Gödel expressed scepticism that it would be easy to formalise what is meant by ‘computation’.

Formalising ‘computation’, 1931

- The formalising of what is meant by ‘computation’ required a precise and convincing definition of *method*. Whatever methods were defined had to be general enough that they could *never be superseded* by a more powerful class of methods.
- Obviously some philosophical, as well as mathematical, analysis was called for.

Turing’s machine, 1936

- The Englishman Alan Mathison Turing provided a solution to Hilbert's *Entscheidungsproblem* by constructing a formal model of a computer,

  and showing that there were problems that such a machine could not solve
  - proving computation (and mathematics) was undecidable!

Turing's machine, 1936

- The 1936 concept of the Turing machine appears not only in mathematics and computer science, but in cognitive science and theoretical biology.

Turing's machine, 1936

- In CCT too, we regularly use Turing's model of computation to evaluate measures of complexity.
- Since almost all effective electronic computer systems today are based upon Turing's (and von Neumann's) vision of a computer, it is important to follow the introduction of his 'machine' in detail.

Turing's machine, 1936

- Familiarity with a particular model of computation will allow us to introduce the necessary modifications or features with which a more efficient algorithm for a known problem could be found.
- Eg: squaring example.

Turing's machine, 1936

- It will be instructive to place Turing's creation in the context of the frontiers of computing at the time.
- A short review of the mathematical mechanical inventions of the previous centuries will suffice, followed by a look at Turing's academic background up to this date.

Mathematical machines, pre-1935

- 80 BC, the Greek Antikythera mechanism, metal gears and pointers predicting the motions of the stars and planets.
- 1612, John Napier, invents logarithms and several machines for multiplication, including a little known chessboard calculator...

Mathematical machines , pre-1935

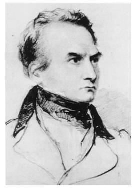
- 1622, William Oughtred, slide rule based on Napier's logarithms, was the primary calculator of engineers through the 19th and early 20th centuries.
- 1623, William Schickard, a simple adder permitting the multiplication of multi-digit numbers.

Mathematical machines , pre-1935

- 1642, Blaise Pascal, the first adding machine with an automatic carry.
- 1673, Gottfried Leibniz, a calculator capable of multiplication in which a number was repeatedly and automatically added into an accumulator.
- 1801, Joseph-Marie Jacquard, an automatic loom using punched cards.

Mathematical machines , pre-1935

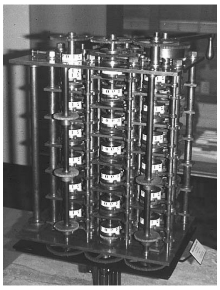
- Charles Babbage is famous for receiving the first government grant for computer research, and of course for designing his Difference Engine.



- In 1822 he first suggested that it might be possible to compute the entries of the common mathematical and navigation tables of the day using a steam engine.

Mathematical machines , pre-1935

- 1833, Babbage recognises the limits of the Difference Engine's single operation and designs the Analytical Engine, which has the basic components of a modern computer.



Difference engine

- 1854, in *An Investigation of the Laws of Thought* George Boole describes his system of symbolic and logical reasoning

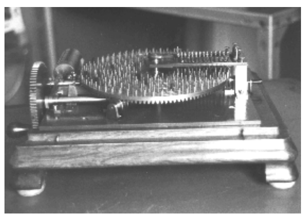
  - thus defining the minimum number of states or symbols required for universal computation (although he didn't appreciate those consequences at the time).

Mathematical machines , pre-1935

- 1869, William Stanley Jevons, England, builds a machine to solve logic problems.
- 1890, punched card data processing machines from Hollerith Tabulating Company (a precursor of IBM) used in the US Census.

Mathematical machines , pre-1935

- 1919, Eugène Olivier Carissan, France, designed and built a mechanical device for factoring integers and testing them for primality.



Mathematical machines , pre-1935

- ~1920, Leonardo Torres y Quevedo, Spain, built some electromechanical calculating devices, including one that played simple chess endgames.
- And finally, 1925, Vannevar Bush of MIT builds the large-scale *'Differential Analyzer'* with integration and differentiation capabilities.

Turing's background

- Turing was born in London and went to King's College, Cambridge.

Turing's background

- His interest in logical foundations was provoked by the English relativity physicist Arthur Stanley Eddington
  - who asserted that quantum mechanics yielded room for human free will, and questioned how a collection of ordinary atoms could be a thinking machine.
- Turing thought that it might be possible to model the brain mathematically, and thus mechanically.

Turing's background

- In 1935, Turing, then 23 years old, attended the advanced lectures on the Foundations of Mathematics given by the Cambridge topologist Maxwell Newman, which brought him up to the point reached by Gödel in his 1931 *Incompleteness theorem.*
- Turing worked by himself for a year and only published when he had fully formulated the concept of *computability*.

Turing's background

- Looking at Turing's machines in a later unit it will become evident that Turing, in his definition of computability, was not considering the computation machines of his day.
- He was modelling the action of the human mind.

Turing, "On computable numbers..."

- A. M. Turing,

  "On computable numbers, with an application to the *Entscheidungsproblem*",

  Proceedings of the London Mathematical Society, 1936