CS133 Lecture 24

A Brief History of Computer Science *(with thanks to Prabhakar Ragde)*

Why History?

- No shortage of good stories
- It helps us understand the way things are
- It helps us deal with the way things might be
- But where to begin?

The Dawn of Computation

Babylonian cuneiform
(circa 2000 B.C)

Early Computation

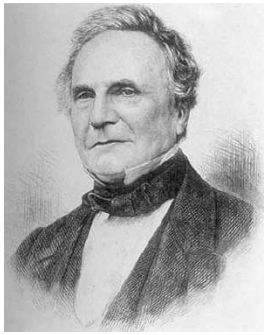
- "Computer" = human being performing computation
- Euclid's algorithm for greatest common divisor, circa 300 B.C.
- Al-Khwarizmi's books on algebra using Hindu-Arabic numerals, circa 800 A.D
- Isaac Newton (1643-1727)

1801: Jacquard loom

- Loom: weaves fabric
- Design of fabric determined by instructions on punched cards
- Specification and execution are separated

Charles Babbage

- England, 1791-1871
- 1819: Difference Engine (machine for tabulating polynomials)
- Babbage's design was too ambitious
- 1834: Analytical Engine (general-purpose computing device)

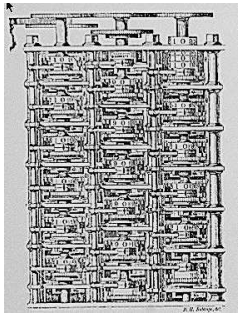


Ada Augusta Byron

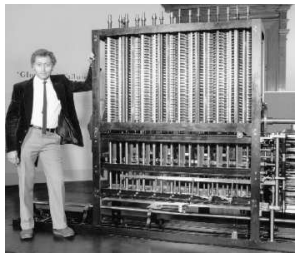
- England, 1815-1852
- Assisted Babbage in explaining and promoting his ideas
- Wrote articles describing operation and use of the Analytical Engine
- The first computer scientist?

Difference Engine

Around 1880



Around 1990



1928: David Hilbert

- Goal of mathematics: to formally prove statements such as “Every even number is the sum of two prime numbers.”
- Proof: finite sequence of deductions from axioms
- Ideally: every statement can either be proved true or false

Hilbert’s questions

- Is mathematics consistent? If so, we can’t prove both a statement and its opposite.
- Is mathematics complete? If so, we can either prove a statement, or prove its opposite. Nothing is unprovable.
- Is there a procedure that can determine the truth or falsity of any mathematical statement?

1930: Kurt Gödel's answers

Surprising answers:

– Mathematics cannot prove itself consistent

– If consistent, mathematics is incomplete (Gödel constructed a statement which is true but has no proof)

Gödel's proofs are brilliant but complicated

After Gödel's proof

- Gödel's unprovable statement was constructed in a fairly artificial way
- Maybe every natural statement is provable true or false?
- Maybe there is a procedure that can determine if a statement is unprovable?

1936: Alan Turing

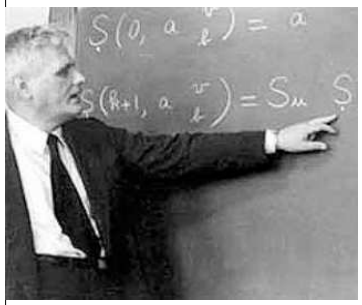
- Turing proved that there is no procedure to determine whether a mathematical statement has a proof.
- How can one prove that such a procedure **doesn't** exist?
  - By precisely defining what "a procedure" is, and showing that every procedure fails to answer the question

Turing's proof

- Turing defined a formal model of computation
- His proof is much simpler than Gödel's
- Key idea: programs that use other programs as data (central to computer science today)
- His proof shows that many natural questions we wish to ask about programs cannot be answered by programs (e.g. do two programs compute the same function?)
- All this before electronic computers even existed!

But he was not the first...!

- Alonzo Church (1903-1995) had already published another "uncomputability" proof
- Church's proof resembled Gödel's, and was more complicated than Turing's proof



The Church-Turing thesis

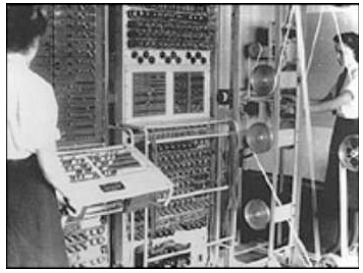
- The two formal models of computation described by Church and Turing were quite different, but proved to be equivalent
- This is true of all "reasonable" models
- Turing's model was more intuitive and influenced the design of hardware and early software
- Church's model was more mathematical, and influenced later software, and methods to formally describe and reason about programming languages

1940: Alan Turing and Enigma

- Enigma: German cipher machine used to encipher radio transmissions
- Encryption was based on a secret key whose discovery would crack the code
- Turing and other British mathematicians were recruited to use statistical methods to discover the secret keys in use

1940-44: Colossus

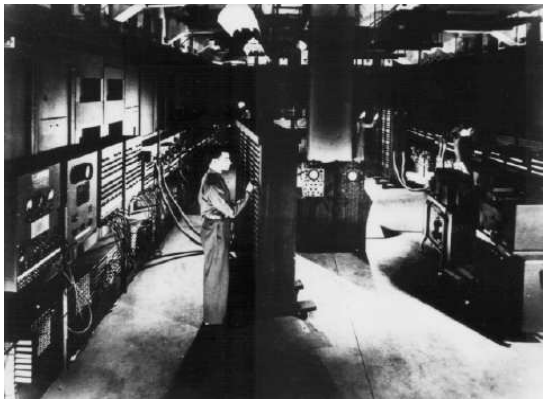
- Series of machines developed to crack Enigma
- Colossus succeeded in deciphering German communications
- It was the world's first electronic general purpose computer



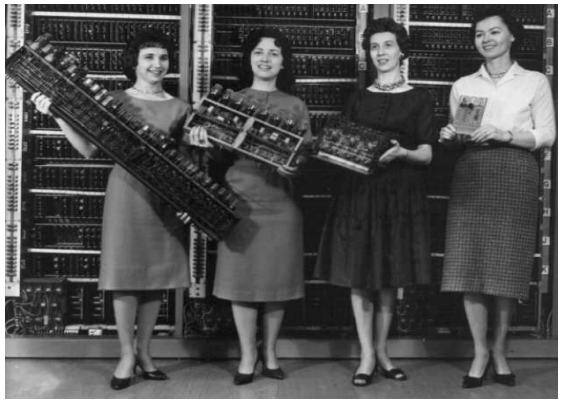
Turing after the war

- Worked at designing computer hardware and software at UK universities
- 1950: Turing test – what would a program have to do to be considered “intelligent”?
- 1954: untimely death at age 42

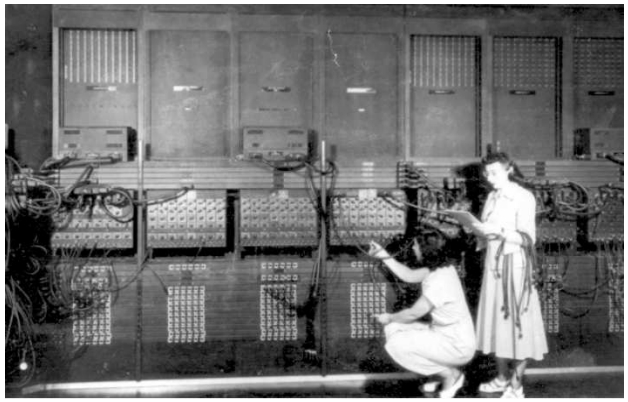
1944-5: ENIAC, U. Penn, USA



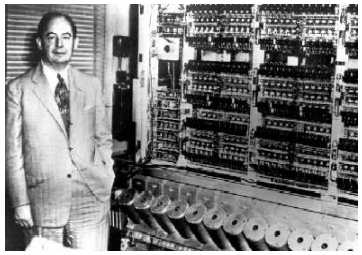
1944-5: ENIAC, U. Penn, USA



1944-5: ENIAC, U. Penn, USA



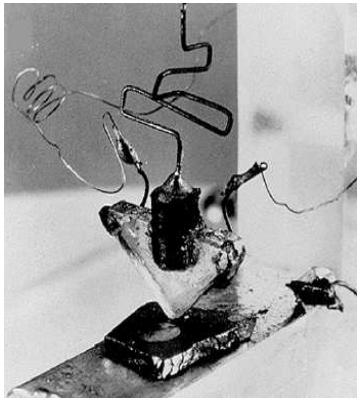
John von Neumann (1903-1957)



- EDVAC report(1946)
- Described key aspects of modern computers (CPU, memory, stored program)

1947: the transistor

- Solid-state switch
- Not fragile, leaky, or slow, like vacuum tubes or relays
- Rapidly miniaturized
- Moore's Law (1965): density of transistors on a chip doubles every 18 months



1950: Grace Murray Hopper

- Wrote first compiler (translated high level language into machine language)
- Key player in standardization of COBOL language (especially English-like syntax)

Grace Murray Hopper, 1906-92

FORTRAN (1957)

```
      DIMENSION A[10]
      I1 = 1
      DO 3,5,6 J=1,10
3     I1 = I1 + I2
      I2 = I1 - I2
5     A[J] = I1
6     STOP
```


LISP (1959)

```
(defun fhelper (n)
 (if (= n 1)
    '(1 1)
     (let (pl (fhelper (- n 1)))
           (cons (+ (car pl)
                 (cadr pl)) pl))))
(defun fib (n)
    (reverse (fhelper n)))
```

ALGOL (1960)

```
begin
  integer prev, curr, j, a[10];
  prev := 0;
  curr := 1;
  for j := 1 step 1 to 10 do
  begin
     curr := curr + prev;
     prev := curr - prev;
     a[j] := curr;
  end
end
```


1950-60's

At this point, computers were large, standalone devices owned by corporations and universities.

They used punched cards and paper printouts, or "dumb" terminals displaying lines of monospaced font like this.

"I think there is a world market for maybe 5 computers." IBM Chair, Thomas Watson.

"Computers in the future may have only 1000 vacuum tubes and perhaps weigh only 1½ tons." Popular Mechanics, 1949

Networks, personal computers and graphical displays changed all that.

1969: ARPAnet

- Network connecting computers separated geographically (with dedicated phone lines)
- 1972: e-mail, syntax userid@machine
- 1974-7: other networks (Tymnet, UUnet)
- 1979: Usenet (newsgroups)
- 1982: Internet Protocol (IP)

1975: Altair

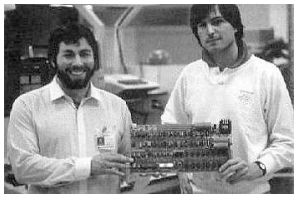
- Microprocessor kit for under $500
- 256 bytes (!) memory
- No keyboard, no monitor, no storage device
- BASIC compiler developed by Bill Gates and Paul Allen

Personal Computers

- 1976: Apple I board
- 1977: complete machines (Apple II, Commodore PET, Radio Shack TRS-80)
- 1981: IBM PC (with DOS supplied by Microsoft)

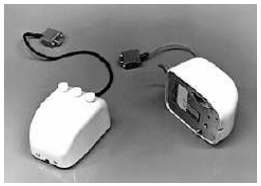
"640K ought to be enough for anybody." Bill Gates, 1981.

Steve Jobs and Steve Wozniak with Apple I (all of it)



Windows and Mice

- 1968: Doug Englebart's mouse (Stanford Research Institute)
- 1972: Xerox Alto (windowing system with mouse)
- 1984: Apple Macintosh (first commercially successful system based on graphical user interface)



1989: The World-Wide Web

- Developed by Tim Berners-Lee at CERN as means of information exchange among scientists

- Ideas from Vannevar Bush (1945), Ted Nelson (1965), Doug Englebart
- 1993: NCSA Mosaic graphical browser developed

Development of Java

- 1991, Sun Microsystems: "The Green Project" – formed to determine the next wave of computing
  - "Digitally controlled consumer devices and computers"

- Group focused on designing tools for the new field of interactive TV
  - That didn't quite work as hoped, but …

Development of Java, continued

- SUN realized the supporting language had larger applications
  the World Wide Web (and more!)
- 1995: Netscape licensed it for its browser
- 1995: SUN released it to the public
  Java (formerly OAK)
  Primary designer:
  James Gosling (of Calgary)

What's next?

- Molecular computing
- Biological computing
- Quantum computing
- Nanotechnology
- Ubiquitous computing
- …?

ERROR: undefined
OFFENDING COMMAND:

STACK: