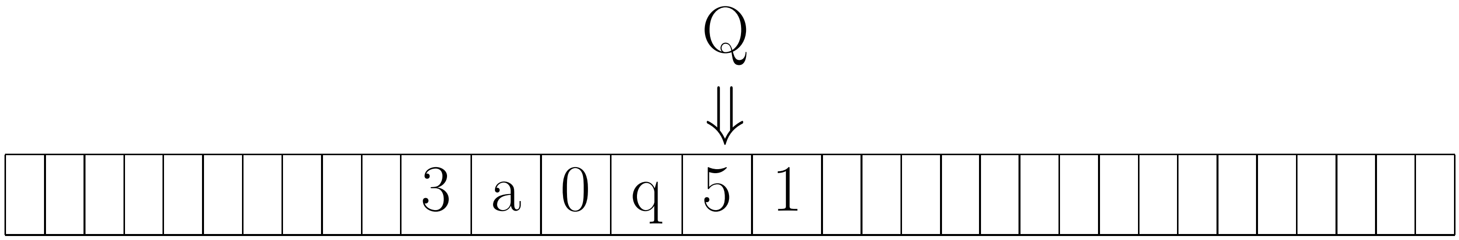
UNIVERSALITY IS UBIQUITOUS

Martin Davis

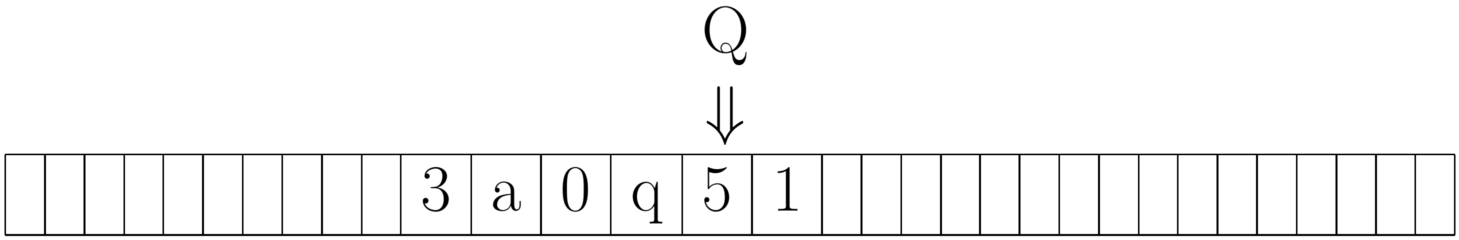
Professor Emeritus
Courant Institute, NYU

Visiting Scholar
UC Berkeley



Turing machine operation:

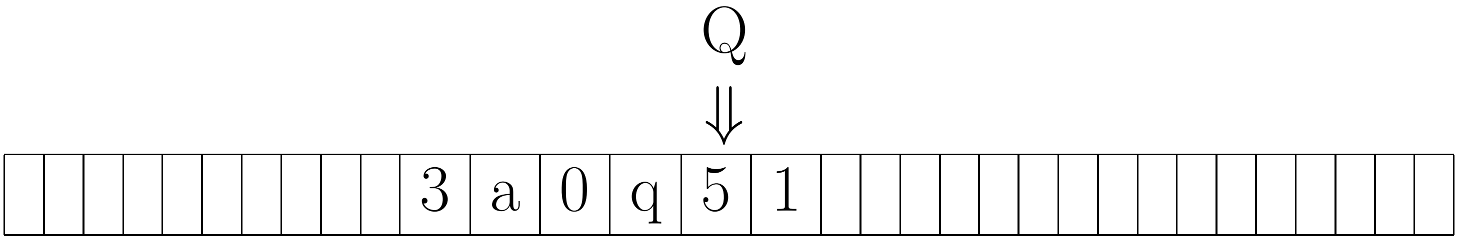
- Replace symbol (”print”)
- Move left or right one square, or don’t move
- Change state



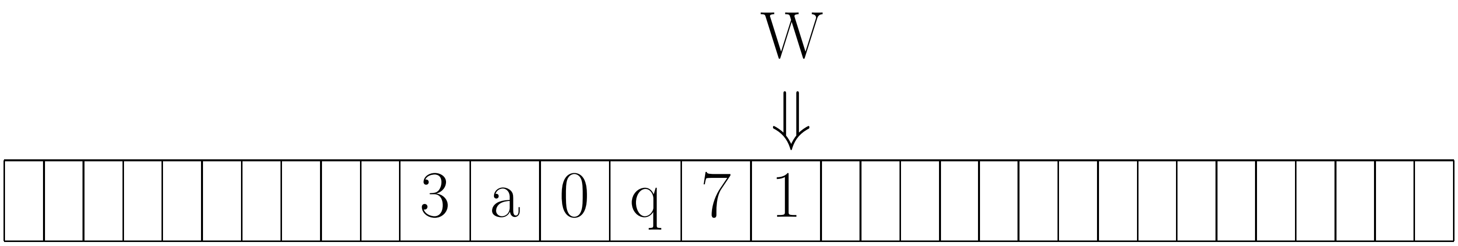
A “quintuple” representing one step in a Turing machine computation: Q5:7→W

Turing machine operation:

- Replace symbol (”print”)
- Move left or right one square, or don’t move
- Change state



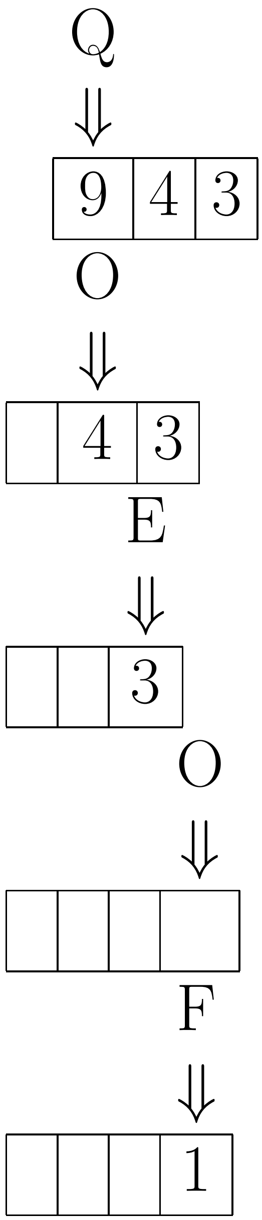
A "quintuple" representing one step in a Turing machine computation: Q5:7→W



Turing machine operation:

- Replace symbol ("print")
- Move left or right one square, or don't move
- Change state

$Q\,0 : \square \longrightarrow E$	$Q\,2 : \square \longrightarrow E$	$Q4 : \square \longrightarrow E$	$Q6 : \square \longrightarrow E$	$Q8 : \square \longrightarrow E$
$Q\,1 : \square \longrightarrow O$	$Q\,3 : \square \longrightarrow O$	$Q5 : \square \longrightarrow O$	$Q7 : \square \longrightarrow O$	$Q9 : \square \longrightarrow O$
$E\,0 : \square \longrightarrow E$	$E\,2 : \square \longrightarrow E$	$E4 : \square \longrightarrow E$	$E6 : \square \longrightarrow E$	$E8 : \square \longrightarrow E$
$E\,1 : \square \longrightarrow O$	$E\,3 : \square \longrightarrow O$	$E5 : \square \longrightarrow O$	$E7 : \square \longrightarrow O$	$E9 : \square \longrightarrow O$
$O\,0 : \square \longrightarrow E$	$O\,2 : \square \longrightarrow E$	$O4 : \square \longrightarrow E$	$O6 : \square \longrightarrow E$	$O8 : \square \longrightarrow E$
$O\,1 : \square \longrightarrow O$	$O\,3 : \square \longrightarrow O$	$O5 : \square \longrightarrow O$	$O7 : \square \longrightarrow O$	$O9 : \square \longrightarrow O$
$E\square : 0 \ \star \ F$	$O\square : 1 \ \star \ F$			



Alan Turing's universal machine:

Code of $\mathcal{M}$		Input to $\mathcal{M}$				

It provides a model of a memory shared between program and data.

What enables universality in a computational setup?

What enables universality in a computational setup?

Could it be a rich supply of basic operations?

What enables universality in a computational setup?

Could it be a rich supply of basic operations?

Hardly! It takes very little in basic operations to achieve universality!

What enables universality in a computational setup?

Could it be a rich supply of basic operations?

Hardly! It takes very little in basic operations to achieve universality!

It’s Infinite memory!!

Howard Aiken (1956):

“If it should turn out that the basic logics of a machine designed for the numerical solution of differential equations coincide with the logics of a machine intended to make bills for a department store, I would regard this as the most amazing coincidence I have ever encountered.”

Alan Turing 1947

(Address delivered to the London Mathematical Society)

Let us now return to the analogy of the theoretical computing machines … It can be shown that a single special machine of that type can be made to do the work of all. It could in fact be made to work as a model of any other machine. The special machine may be called the universal machine …

Turing 1947

I considered a type of machine which had a central mechanism, and an infinite memory which was contained on an infinite tape. Machines such as the ACE may be regarded as practical versions of ... the type of machine I was considering ... There is at least a very close analogy ... digital computing machines such as the ACE ... are in fact practical versions of the universal machine.

From von Neumann (et al) 1946

It is easy to see by formal-logical methods that there exist codes that are *in abstracto* adequate to control and cause the execution of any sequence of operations … The really decisive considerations from the present point of view, in selecting a code, are of a more practical nature: simplicity of the equip-ment demanded by the code, and the clarity of its application to the actually important problems together with the speed of its handling those problems. It would take us much too far afield to discuss these questions at all generally or from first principles.

Turing's ACE and von Neumann's EDVAC

Von Neumann's primary interest in numerical calculation was reflected in the logical organization of the EDVAC and its successors (H-bomb calculations).

- ACE to be used for many tasks for which heavy arithmetic was inappropriate (chess)
- ACE was organized in a more minimal way with arithmetic operations to be carried out by software rather than hardware.
- The ACE design provided special mechanism for incorporating previously programmed operations in a longer program.

Turing on a proposal to modify the ACE in a von Neumann direction:

> [It] is ... very contrary to the line of development here, and much more in the American tradition of solving one's difficulties by means of much equipment rather than by thought. ... Furthermore certain operations which we regard as more fundamental than addition and multiplication have been omitted.

From Davis 2000

Before Turing the … supposition was that …the three categories, machine, program, and data, were entirely separate entities. The machine was a physical object … hardware. The program was the plan for doing a computation … [T]he data was the numerical input. Turing's universal machine showed that the distinctness of these three categories is an illusion. A Turing machine is initially envisioned as a machine …, *hardware*. But its code … functions as a *program*, detailing the instructions to the universal machine … Finally, the universal machine in its step-by-step actions sees the … machine code as just more *data* to be worked on. This fluidity …is fundamental to contemporary computer practice. A *program* …is *data* to the … compiler.

Pre-1946 Calculators:

Could they reasonably be called universal?

Pre-1946 Calculators:

Could they reasonably be called universal?

The Analytic Engine (Robin Gandy)

Pre-1946 Calculators:

Could they reasonably be called universal?

The Analytic Engine (Robin Gandy)

The Colossus (Benjamin Wells)

Pre-1946 Calculators:

Could they reasonably be called universal?

The Analytic Engine (Robin Gandy)

The Colossus (Benjamin Wells)

The Einiac (Arthur & Alice Burks)

Pre-1946 Calculators:

Could they reasonably be called universal?

The Analytic Engine (Robin Gandy)

The Colossus (Benjamin Wells)

The Einiac (Arthur & Alice Burks)

Enough basic operations?

Pre-1946 Calculators:

Could they reasonably be called universal?

The Analytic Engine (Robin Gandy)

The Colossus (Benjamin Wells)

The Einiac (Arthur & Alice Burks)

Enough basic operations?

Yes

Pre-1946 Calculators:

Could they reasonably be called universal?

The Analytic Engine (Robin Gandy)

The Colossus (Benjamin Wells)

The Einiac (Arthur & Alice Burks)

Enough basic operations?

Yes

Approximating infinite memory?

Pre-1946 Calculators:

Could they reasonably be called universal?

The Analytic Engine (Robin Gandy)

The Colossus (Benjamin Wells)

The Einiac (Arthur & Alice Burks)

Enough basic operations?

Yes

Approximating infinite memory?

NO!

THE AI HYPOTHESIS

1. The human mind is a function of the brain.
2. The brain is a computer.

THE AI HYPOTHESIS

1. The human mind is a function of the brain.
2. The brain is a computer.

If you believe this, the AI project is reasonable.

THE AI HYPOTHESIS

1. The human mind is a function of the brain.
2. The brain is a computer.

If you believe this, the AI project is reasonable.

If you reject the **possibility** of human level AI, you should reject 1 or 2.

1. The human mind is a function of the brain.
2. The brain is a computer.

Roger Penrose believes 1 but not 2. He argues that 2 implies that our mathematical capabilities would be limited by Gődel incompleteness, whereas we can **see** that the Gődel undecidable statement is true.

1. The human mind is a function of the brain.
2. The brain is a computer.

John Searle wrote:

“Here is what is going on inside Deep Blue. The computer has a bunch of meaningless symbols that the programmers use to represent the positions of the pieces on the board. It has a bunch of equally meaningless symbols that the programmers use to represent options for possible moves. The computer does not know that the symbols represent chess pieces and chess moves, because it does not know anything.”

More from Searle:

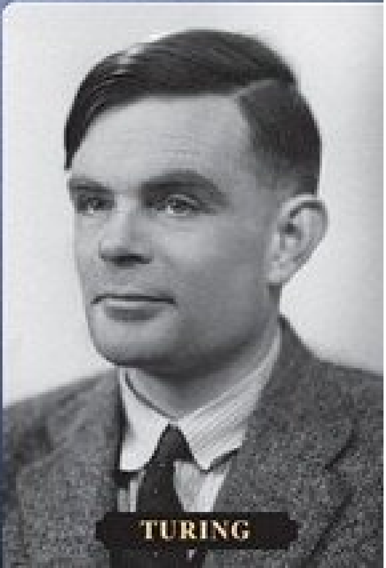
Imagine that a man who does not know how to play chess is locked inside a room, and there he is given a set of, to him, meaningless symbols. Unknown to him, these represent positions on a chessboard. He looks up in a book what he is supposed to do, and he passes back more meaningless symbols. We can suppose that if the rule book, i.e., the program, is skillfully written, he will win chess games. People outside the room will say, ``This man understands chess, and in fact he is a good chess player because he wins.'' They will be totally mistaken. The man understands nothing of chess, he is just a computer. And the point of the parable is this: if the man does not understand chess on the basis of running the chess-playing program, neither does any other computer solely on that basis.

the Universal Computer

Turing Centenary Edition

The Road from Leibniz to Turing

Martin Davis



CRC Press
Taylor & Francis Group
AN A K PETERS BOOK