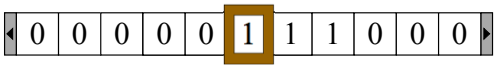
Eric Roberts
CS 54N

Handout #12
October 19, 2016

Turing Machines

Turing Machines



Eric Roberts
CS 54N
October 19, 2016

Mathematics Enters the 20th Century

If we would obtain an idea of the probable development of mathematical knowledge in the immediate future, we must let the unsettled questions pass before our minds and look over the problems which the science of today sets and whose solution we expect from the future.


David Hilbert (1862-1943)

- In 1900, the eminent German mathematician David Hilbert set out a series of challenging problems for mathematicians in the twentieth century.
- Many of those problems turned out to be relatively easy, but several remain unsolved even today.
- Several of Hilbert's 23 original problems, along with others he devised later, were resolved in a way that shook the foundations of the mathematical community.

Hilbert's *Entscheidungsproblem*

- Of the problems that have significance to computer science, the most important is the *Entscheidungsproblem*, which was posed in 1928. In informal terms, the *Entscheidungsproblem* can be expressed as follows:

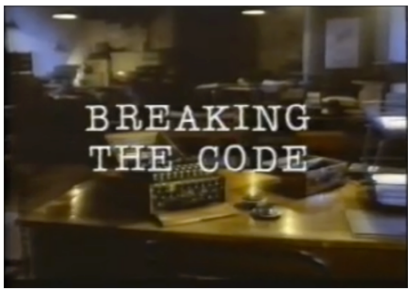
  Is it possible to find a mechanical procedure that can determine, given a specific proposition in a formal system of symbolic logic, whether that proposition is provable within that system?

- Hilbert and the mathematicians of his day assumed that such a mechanical procedure was indeed possible, but a series of mathematical results in the 1930s—by Kurt Gödel, Alonzo Church, Alan Turing, and Emil Post—showed that such a mechanical procedure is a logical impossibility.

Alan Turing's Contribution

- Alan Mathison Turing, a young British mathematician just out of Cambridge, helped settle the *Entscheidungsproblem* by developing a model for computation by a mechanical procedure.
- Turing's model—which is now known as a ***Turing machine***—is a central concept in theoretical computer science.
- Turing is widely recognized as one of the most important figures in the history of computer science. The field's most prestigious prize is the Turing Award, which is given in his honor.


Alan Turing (1912-1954)



Designing the Turing Machine

- In his groundbreaking 1936 paper, "On computable numbers, with an application to the *Entscheidungsproblem,*" Turing described the process of computation in informal terms:

> Computing is normally done by writing certain symbols on paper. We may suppose this paper is divided into squares like a child's arithmetic book. In elementary arithmetic the two-dimensional character of the paper is sometimes used. But such a use is always avoidable, and I think it will be agreed that the two-dimensional character of paper is no essential of computation. I assume then that the computation is carried out on a one-dimensional paper, i.e. on a tape divided into squares. I shall also suppose that the number of symbols which may be printed is finite. . . .

Designing the Turing Machine

- In his groundbreaking 1936 paper, "On computable numbers, with an application to the *Entscheidungsproblem,*" Turing described the process of computation in informal terms:

> The behaviour of the computer at any moment is determined by the symbols which he is observing, and his "state of mind" at that moment. We may suppose that there is a bound B to the number of symbols or squares which the computer can observe at one moment. If he wishes to observe more, he must use successive observations. We will also suppose that the number of states of mind which need be taken into account is finite. . . .
>
> The simple operations must therefore include:
>
> (a) Changes of the symbol on one of the observed squares.
>
> (b) Changes of one of the squares observed to another square within L squares of the previously observed squares.

Designing the Turing Machine

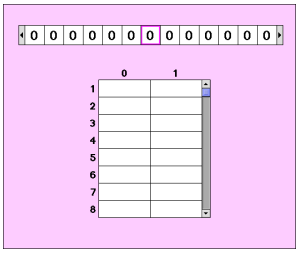
- In his groundbreaking 1936 paper, "On computable numbers, with an application to the *Entscheidungsproblem,*" Turing described the process of computation in informal terms:

> It may be that some of these changes necessarily involve a change of state of mind. The most general single operation must therefore be taken to be one of the following:
>
> (a) A possible change of symbol together with a possible change of state of mind.
>
> (b) A possible change of observed squares, together with a possible change of state of mind.
>
> The operation actually performed is determined, as has been suggested above, by the state of mind of the computer and the observed symbols.

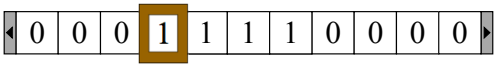
- These operations and the notion of a "state of mind" form the basis for the Turing machine.

Turing Machine Components

Computation requires:
- Scratch paper
- An unbounded amount of space
- At least two symbols
- A read/write mechanism
- Some form of program control

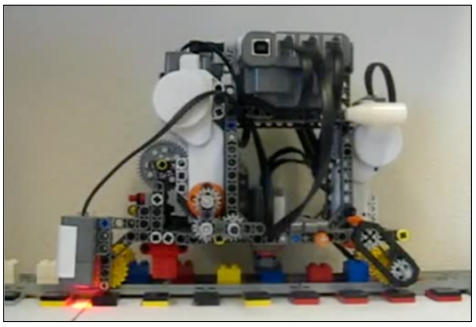


A Sample Turing Machine



	0	1
1	1R2	1L2
2	1L1	1L0

The Lego Turing Machine



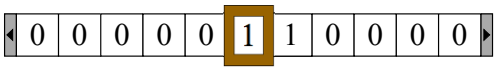
Representing Numbers

- Even though the standard Turing machine alphabet consists of the digits **0** and **1**, it is not practical to represent numbers in binary. *Why?*
- Instead, numbers will be written in ***unary*** in which each number is written as a sequence of **1**s. The **0** symbol is used to indicate the start and end of a number.
- An input configuration for the Turing machine is ***well-formed*** if it consists of a single number in which the tape head appears over the first **1** digit.
- A Turing machine program is a ***function*** if it starts with one well-formed number and ends with a well-formed number.

The Add3 Function (M_{+3})

	0	1
1	1L2	1L1
2	1L3	
3	1R3	1L0

Try it with an input value of 2:

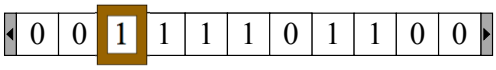


The Doubler Function (M_{2x})

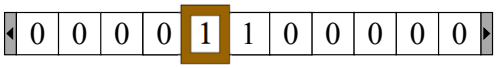
	0	1
1	0R0	0R2
2	0R3	1R2
3	1R4	1R3
4	1L5	
5	0L6	1L5
6	0R1	1L6

Exercise: Subtraction

Write a program that takes two numbers on the tape and subtracts the second from the first. Thus, if the initial tape contains



the final tape should look like this:



Assume for the moment that the first number is larger than the second. What happens to your program if that isn't true?

Composing Machines (M_{2x+3})

Suppose you wanted to compute the function $2x + 3$, given that you have the machines M_{2x} and M_{+3}.

- Start with the two machines.

	0	1
1	0R0	0R2
2	0R3	1R2
3	1R4	1R3
4	1L5	
5	0L6	1L5
6	0R1	1L6

M_{2x}

1	1L2	1L1
2	1L3	
3	1R3	1L0

M_{+3}

Composing Machines (M_{2x+3})

Suppose you wanted to compute the function $2x + 3$, given that you have the machines M_{2x} and M_{+3}.

- Start with the two machines.
- Renumber the states in M_{+3}.
- Combine the machines.
- Change halt transitions in M_{2x} to jump to M_{+3}.

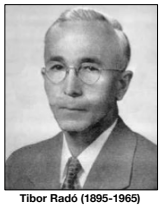
	0	1
1	0R7	0R2
2	0R3	1R2
3	1R4	1R3
4	1L5	
5	0L6	1L5
6	0R1	1L6
7	1L8	1L7
8	1L9	
9	1R9	1L0

M_{2x+3}

The Busy Beaver Problem

- Although it is possible to introduce the notion of undecidable problems using Turing's original argument involving a "universal" Turing machine, it is much easier to do so in the context of a more recent problem posed by Tibor Radó in the early 1960s:

  What is the largest finite number of **1**s that can be produced on blank tape using a Turing machine with *n* states?



Tibor Radó (1895-1965)

- This problem is called the ***Busy Beaver Problem***.