Lecture 22: Turing Machines

MATH230

Te Kura Pāngarau | School of Mathematics and Statistics
Te Whare Wānanga o Waitaha | University of Canterbury

Outline

Summary

In 1931 Gödel proved his incompleteness theorems. These gave *negative* answers to some of the metaquestions that Hilbert had asked. The question of decidability remained open. Negative answers would have been easier to entertain in the light of Gödel.

Although important for the development of the theory of computation, primitive recursion and μ-recursion were not used to resolve the effective calculability metaquestions. It was papers on other definitions of computation that settled the entscheidungsproblem and the undecidability of arithmetic.

Both papers appeared in 1936 at about the same time:

Turing Machines (Alan Turing)

λ-calculus (Alonso Church)

Turing went to do his PhD with Church *after* these were published.

Alan Turing, 1936

In 1936 Alan Turing published the paper *On Computable numbers, with an application to the entscheidungsproblem.* In this paper he presented his reply to the hope for a definition of an effective procedure.

He developed the theory of $\mathfrak{a}$-machines and used them to answer the entscheidungsproblem in the negative.

Turing called his machines automatic ($\mathfrak{a}$) machines. It was his thesis advisor, Church, that coined the term Turing Machine.

Finite Means

Turing opened his paper as follows:

> *Computable numbers may be described briefly as the real numbers whose expressions as a decimal are calculable by finite means.*

Turing did not mean to say the number itself had to be expressed as a finite decimal. But that the means for calculating it were finitely expressed.

This sets his perspective apart from that of primitive recursion (and λ-calculus) by focussing on the number, rather than functions on numbers. He quickly points out that it is just a change in perspective and that his approach can equally be considered from the point-of-view of functions.

Step-by-Step Calculation

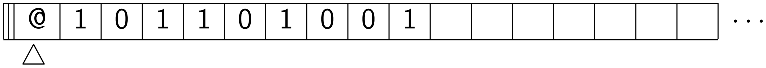
We may compare a man in the process of computing a real number to a machine which is only capable of a finite number of conditions $q_1, q_2, \ldots, q_R$ which will be called m-configurations.

$$\begin{array}{cccccc} 1 & 0 & 1 & 1 & 0 & 1 \\ & & & + & 1 & 0 \\ \hline \end{array}$$

+	0	1
0	0	1
1	1	0

a-Machines

The machine is supplied with a "tape" (the analogue of paper) running through it, and divided into sections (called "squares") each capable of bearing a "symbol"



Turing Machine

Tape

Tape has a start (`@`) cell.

Tape is infinite to the right.

Tape squares hold all input, output, and working.

Read Head

Scan one square at a time; scanned square.

Can move left, nowhere, right (L/N/R).

Prints symbols to a square.

$\mathfrak{m}$**-configurations** (q_1, q_{12}, q_{13} etc.)

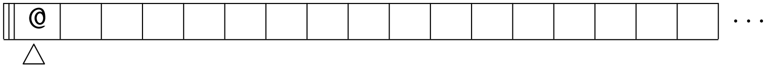
Finitely many configurations.

Print/Move/Update all defined by configurations.

How to react to the scanned symbol.

Start in q_0. Special *HALT* state.

Example: Boring

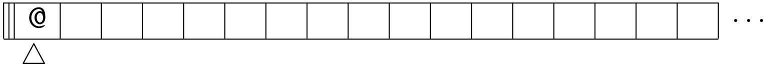


C	R	P	M	U
q_0	@	@	R	H

Current State:	q_0
Scanned Symbol:	@
Printed Symbol:	@
Move Direction:	R
Update State:	H

Example: All Ones n Zeroes

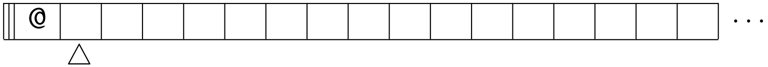
Write instructions for a Turing machine that will not HALT and prints an alternating pattern of 1s and 0s.



C	R	P	M	U
q_0	@	@	R	

Current State: q_0
Scanned Symbol: @
Printed Symbol: @
Move Direction: R
Update State:

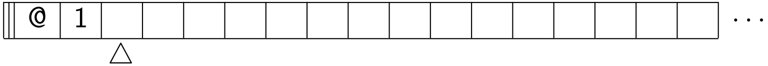
Example: All Ones n Zeroes



C	R	P	M	U
q_0	@	@	R	q_0
q_0	b	1	R	q_1
q_1	b	0	R	q_0

Current State:	q_0
Scanned Symbol:	b
Printed Symbol:	1
Move Direction:	R
Update State:	q_1

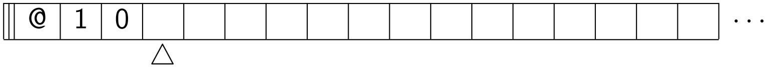
Example: All Ones n Zeroes



C	R	P	M	U
q_0	@	@	R	q_0
q_0	b	1	R	q_1
q_1	b	0	R	q_0

Current State:	q_1
Scanned Symbol:	b
Printed Symbol:	0
Move Direction:	R
Update State:	q_0

Example: All Ones n Zeroes



C	R	P	M	U
q_0	@	@	R	q_0
q_0	b	1	R	q_1
q_1	b	0	R	q_0

Current State:	q_0
Scanned Symbol:	b
Printed Symbol:	1
Move Direction:	R
Update State:	q_1

Specification and Execution

To specify a Turing machine is to write down the table of states. Turing eventually converges on the following standard which we will follow in this course:

C	R	P	M	U
q_0	@	@	R	q_0
q_0	b	1	R	q_1
q_1	b	0	R	q_0

$q_5, 1, 0, L, q_{10}$

The execution of the Turing machine consists of starting at the q_0 state, carrying out the instructions of the current state (in the order presented), followed by the instructions of the following states.

The execution loop is only stopped if the special HALT state is reached at some point in the execution of the machine instructions.

Conventions

The following is a list of things to remember

Each instruction has a print command.

Each instruction has a move command.

Special character *b* means *blank square*.

Always HALT on the home (`@`) symbol.

Function input next to `@` follow by blanks

Function output next to `@` followed by blanks

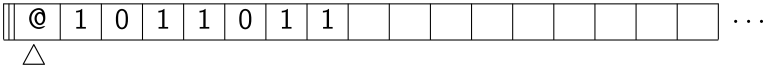
Use non-numerical symbols for working

Give a high-level description of every machine you write.

Comment your machine instructions!

Example: Flip-bits

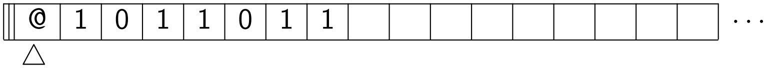
Write a Turing machine to flip the bits on a binary input.



C	R	P	M	U
q_0	@	@	R	

Example: Reverse

Write a Turing machine to reverse a binary input.



C	R	P	M	U
q_0	@	@	R	

Further Reading

Here are some recommended reading to follow up on the lecture content.

- The Annotated Turing *Charles Petzold*
- SEP: Turing Machines