TURING MACHINES, UNDECIDABILITY, AND CRYPTOGRAPHY

L. GEEL

1. Introduction

Turing machines, named after famous British mathematician Alan Turing (1912-1952), is a mathematical computational model which manipulates symbols on a strip of tape according to a table of rules. A Turing machine consists of an infinite piece of tape which is used as the memory. The tape is divided into cells and there's a tape head which points to the currently inspected cell. Finally, there's a state transition table which governs the behavior of the machine. Each cell of the tape can have one of a predetermined finite set of symbols, one of which is the blank symbol. While this might seem simple enough, you can construct a Turing machine given any computer algorithm and such machine will be capable of implementing that algorithm's logic no matter how complex the algorithm. There have been many uses of Turing machines throughout history, many of which involve cryptography and decryption.

2. Basic Functions of Turing Machines

The interesting thing about Turing machines is that they can be incredibly complex or very rudimentary, it all depends on the algorithm it uses and what it functions as. In simple terms, the functions act as such:
1. What cell I am in right now
2. What value I just read
3. What value I should write
4. What cell I should change to next
5. Which direction I should move in next

To better understand how this works lets look at a basic functioning Turing machine which checks to see if an input is a palindrome.

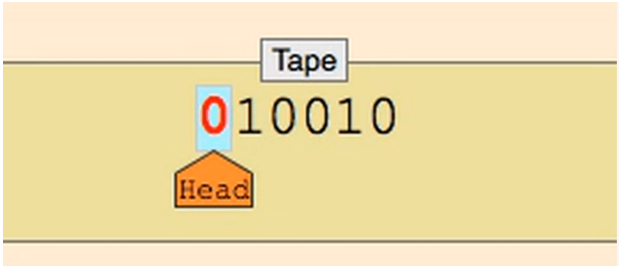

Figure 1. This Turing machine checks if an input string is a palindrome. [KM22]

This Turing machine works like this: The tape head starts on the left most cell and learns its value. Then the tape head moves to the right most cell and learns its value. If the left most and right most cells have the same value the tape head removes the right most cell then moves back to the left most cell and removes that one. It then repeats those steps and if no cells are left then the

Date: April 30, 2022.

input is a palindrome. So in the figure, the 010010 will be reduced to 1001 after the first iteration, 00 after the second, and then be empty after the third and the function will conclude that it is in fact a palindrome.

3. Turing machine in WWII

If you've ever seen the 2014 Oscar-nominated movie The Enigma Machine then you are already somewhat familiar with Turing Machines. This movie was centered around Alan Turing and how his machine was able to help end World War 2. During the war, the Germans used an enciphering machine to send messages securely. The machine would encrypt messages following a rule that was changed daily by the Germans in order to make them impossible to crack. However, thanks to Alan Turing's invention, his machine was able to decipher the code which gave vital information to the Allies. Many people credit this machine as the key factor in the Allies' victory over the Axis.

Figure 2. Alan Turing alongside his machine during WWII [Ire12]

4. Formal Definition

A Turing machine is formally defined as a 7-tuple [KM22] $\langle \mathbb{Q}, q_0, F, \Gamma, b, \Sigma, \delta \rangle$ where
$\mathbb{Q}$ is a finite, non-empty set of states
$q_0 \in \mathbb{Q}$ is the initial state
$F \subset \mathbb{Q}$ is the set of accepting states
Γ is a finite, non-empty set of tape symbols
$b \in \Gamma$ is the blank symbol
$\Sigma \subset \Gamma \setminus \{b\}$ is a set of input symbols
$\delta : (\mathbb{Q} \setminus F) \times \Gamma \to Q \times \Gamma \times \{L, R\}$ is the transition function, which is a partial function

A Turing machine's operation is a sequence $(\langle q'_s, h_s, (t_{s,p})_{p=-\infty}^{\infty} \rangle)_{s=0}^{\infty}$
q_s is the state of the tape head in the s-th step
h_s is the position of the tape head in the s-th step
$t_{s,p}$ is the content of the tape in the s-th step and p-th position
such that:
$q'_0 = q_0$

$h_0 = 0$
If the input string is $(x_1, \ldots, x_n) \in \Sigma^n (x$, then $t_{0,p} = x_{p+1}$ for $0 \leq p \leq n-1$ and $t_{0,p} = b$ otherwise
If $\delta(q_s, t_{s,h_s}) = (q, t, d)$ then:
$q_{s+1} = q$
$h_{s+1} = h_s - 1$ if $d = Ld = L$, and $h_{s+1} = h_s + 1h + 1ifd = Rd = R$
$t_{s+1,p} = tifp = h_s$, and $t_{s+1,p} = t_{s,p}$ otherwise

Otherwise, if $q_s \in Fq$ is not defined, then the machine halts; the rest of the sequence is not defined, and s is the number of steps taken by the machine.

5. Decidability and The Halting Problem

In logic, a true/false decision problem is decidable if there exists an effective method for deriving the correct answer. The halting problem is a decision problem about properties of computer programs on a fixed Turing-complete model of computation, i.e., all programs that can be written in some given programming language that is general enough to be equivalent to a Turing machine. The problem is to determine, given a program and an input to the program, whether the program will eventually halt when run with that input? Think about this python code:

```
x = input()
while x:
  pass
```

If the input is empty, the program will terminate and the answer to the question is: yes, this program will terminate, and if the input isn't empty, the program will loop forever and the answer is: no, this program will not terminate. Using his machine, Alan Turing proved that there can't exist an algorithm that always correctly decides whether, for any given program and input, the program halts when run with that input. The essence of Turing's proof is that any such algorithm can be made to contradict itself and therefore cannot be correct. [Dav65]

6. Conclusion

The Turing Machine is hands down one of the most interesting things I've ever learned about. Alan Turing's machine is said to be on par with almost any modern day computer despite being roughly 100 years old. The Church-Turing thesis states that any computable problem can be solved by a Turing machine. This means that a computer more powerful than the Turing machine isn't required to solve computational problems, essentially making it equally as powerful. This same machine is credited as by a driving force behind the end of WWII and has so many different uses in so many different fields. There's even a whole movie about it!

References

[Dav65] Martin Davis. *The Undecidable, Basic Papers on Undecidable Propositions, Unsolvable Problems And Computable Functions.* New York: Raven Press, 1965.
[Ire12] Corydon Ireland. Alan turing at 100, 2012.
[KM22] Christopher Williams Karleigh Moore, Ivan Koswara. Turing machines, 2022.

1

Department of Mathematics and Statistics, University of Massachusetts, Amherst