IOSR Journal of Computer Engineering (IOSR-JCE)
e-ISSN: 2278-0661,p-ISSN: 2278-8727, Volume 26, Issue 3, Ser. 3 (May. – June. 2024), PP 24-29
www.iosrjournals.org

Turing Machine And Variants

Dr. Hieu D. Vu
Fort Hays State University
600 Park Street, Hays, Ks. 67601

Abstract
In the areas of computing theories: automata, computability and complexity, a question arose in the 1930s. What are the fundamental capabilities and limitations of computers? In 1936, Alan Turing proposed a powerful computing model called Turing machine. It is an abstract computing machine for a general purpose computer, and like a general purpose computer, it cannot solve a certain complex problems. A Turing machine has unlimited memory using a tape to store characters or symbols and blank characters (empty spaces). It uses a head to read or write (store) information on the tape.
The Turing machine is an abstract, simplest computing device but it can solve most problems and provides a fundamental concept for the design of real world computers. It was first introduced in the 1930 by Alan Turing, therefore the name of the machine is Turing, since thenseveral variants of the Turing machine were developed for the purpose of improving the computing capability.
Keywords: *Automata, Computing Theories, Formal language, Science, Technology, Computing Machines, Turing Thesis, Turing machines, Mathematics.Complexities, Computability, Limitations, Memory.*

Date Of Submission: 26-05-2024 Date Of Acceptance: 06-06-2024

I. Introduction

In the areas of theories of computation: Automata, Computability, and Complexity, there is a fundamental question. What are the fundamental capabilities and limitations of computing machines (computers)? That question arose in the 1930s. Since then, technological advances have increased our ability of computing, and brought this question into reality, the world of practical concern.

Automata theory is the starting point for studying theory of computing, it allows practice with formal definitions of computation. Theories of computability and complexity required a precise definition of a computer (what is the capability of the machine?).

A Turing machine is an abstract computing machine that is based on mathematical computation and encapsulated the fundamental logic principles of stored program to be executed in an all-purpose digital computer. Turing machine is a powerful model for computing, first proposed by Alan Turing in 1936 therefore it is called Turing machine. As real computers, Turing machine cannot solve certain problems, which are beyond the theoretical of computing and limitation of computation. Turing machine uses an unlimited tape (memory) to store characters or symbols. Initially, it contains the input string that to be computed (processed), and blanks. Turing machine has a head that can read information or to write (store) information on the tape. [1]

Turing machine is a powerful tool in the study of theories of computing, and it can be used in some applications such as:

- Algorithmic Information Theory and Complexity studies
- Software Testing and Engineering
- High Performance Computing
- Machine Learning and Artificial Intelligence
- Computer Networks [2]

This research paper provides a review of Turing machine and explores its variants.

II. Turing Machine

DESCRIPTION: A Turing machine is a mathematical model to describe an abstract machine that processes characters or symbols on a tape according to some rules. This Turing abstract machine model is simple but capable of implementing mathematical algorithm. It has unlimited memory, processes data, information on tape (also abstract such as IBM tape, cassette tape…). The memory tape is divided into discrete cells that each can store a character or symbol from a finite set of alphabet of the machine.

The head of the Turing machine, at any point during operation it positioned over one of the tape's cells, and the machine would be in one of the state from a finite set of states. And the head will read the symbol in the cell (these characters or symbolsand blanks are input to the Turing machine for processing). Then based on the input symbol and the current state of the machine, it will write on the same current cell a symbol (from the finite set of alphabet) then move the head to the next cell in the left or right direction, or stop the processing. The decision to write what symbol into the current cell and the direction of moving the head (left or right) base on the combination of the current state, the symbol read and a finite set (table) of rules. Like a real computer run into an infinite loop, the Turing machine is possible running into the same situation, moving the head back and forth that never halt.

The infinite loop that Turing machine might encounter proves the existence of limitations on the power of mechanical computation. Turing machine is in fact the simplest design machine that makes it too slow for computing. The real computers are based on different designs use random access memory, that mean the next cell in the sequence could be anywhere in the main memory of the computer systems, to speed up the processing time and avoiding the system locked up (infinite loop).

OVERVIEW: Turing machine is considered an ideal model of a Central Processor Unit (CPU) that controls and processes data on memory system of a computer. Turing machine processes data in the same fashion with canonical computers which process data sequentially. Typically, data stored in this way is using tape, on which the machine can read or write data.

In the context of formal language, automaton aspect, Turing machine can enumerate some arbitrary subset of valid strings of alphabets defined by the automaton. A set of strings created in this enumerating way is called Recursive Enumerable Language. In this particular case, the Turing machine can be used to validate input strings.

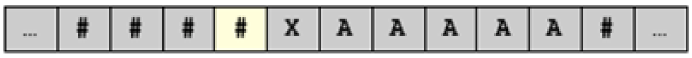
A Turing machine that can simulate other Turing machines is called Universal Turing Machine (UTM) or just simple Universal Machine (UM). Another famous mathematician, computer scientist Alonzo Church (1903-1995) works intertwined with Turing to form the basis for the Church-Turing Thesis which states that Turing machine, Lambda Calculus, and other similar formal computations need to capture information of “effective methods in logic and mathematics” and that will provide a reason about an algorithm or “mechanical procedure”. Studying the abstract properties of Turing machine, we can get an insight into computer science, computability theory and complexity theory. [3]

III. How Does Turing Machine Work?

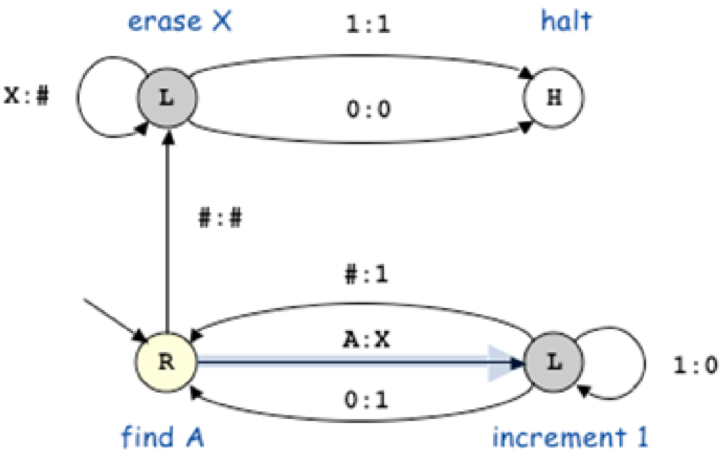
COMPONENTS: Turing machine is the simplest, most primitive model abstract machine that had basic capabilities like human “computer” The machine consists of components below:

The storage (memory) tape that stores the input, intermediate results, and the output. It is long (unlimited memory) and divided into cells. Each cell can store one of a finite alphabet of symbols. In this example the finite set of alphabet includes: 0, 1, A, X, and #

The read/write head of the Turing machine scan the tape one cell at a time. The cell being scanned is the active cell, and it contains the input symbol. At each step, the head reads the input symbol and leave it as is or overwrite a new symbol on it. Then the head will move one position to the left or right. In the example below, the symbol A is replaced by an X, and the move one position to the left.



The control unit is equivalence to the CPU of the real world microprocessor. It consists of a state transition diagram, which is a set of finite instructions that specifies what action the machine to do at each step. Each state represents one of the possible configurations of the machine, depending on the current state and he input symbol. Then the Turing machine may write a new symbol on the current (active) cell, and move to the new state. Each transition connects to one state, for example the current state is s, to another state t, and labeled with two symbols, for example A and X. This example means, if the Turing machine is in state s, and the input symbol is A then overwrite A with X, then transitions (moves) to state t. Each state is labeled with one of the five designations: L(left), R(right), Y(yes), N(no), or H(halt).

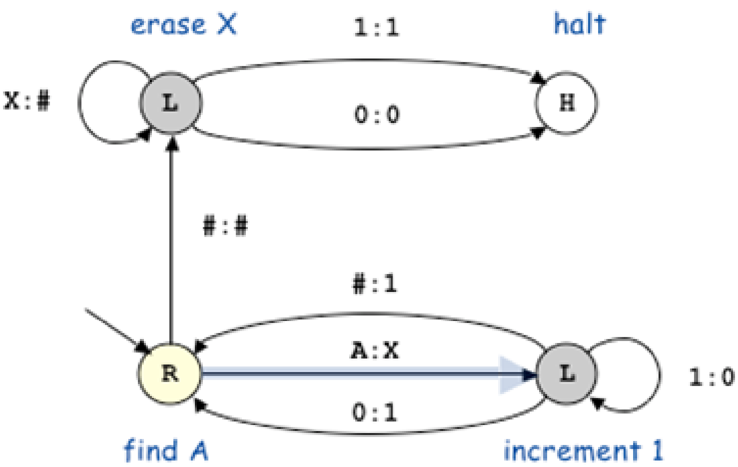


EXECUTION (PROCESSING): Initially, Turing machine starts at the start state, and the read/write head is over a particular cell called start cell. According to the current state and the input symbol, there is at most one possible transition (or do nothing). Each step in a Turing machine proceeds as following:

- Read the input symbol from the active cell (current cell).
- Look up the transition function table for the transition rule associated with the current state and input symbol.
- Write a new symbol into the current cell, to replace the input symbol.
- Change the current state according to the transition rule.
- Move the read/write head one position to the left or right, according to the new state designation.

The steps above is repeated until the current state is labeled H (halt), and the machine will answer Y (yes) or N (no). It is possible the machine will keep running forever such as a real computer run into an infinite loop, without reaching terminating state H (halt).

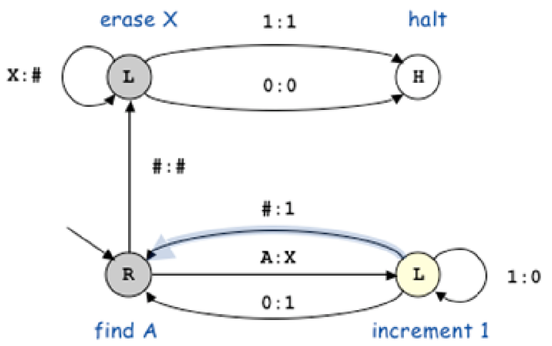
Example: Unary To Binary Conversion:



...	#	#	#	#	A	A	A	A	A	A	#	...

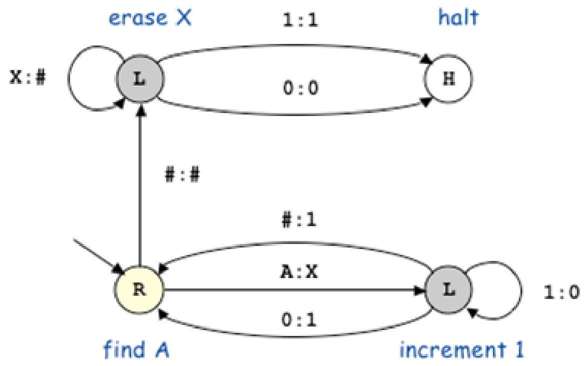
As an example, the Turing machine above has four states: small circles (L, H, R, L). The current state and input symbol are in yellow color (state R and input symbol A). Let trace the execution.

Since the current state is R (right), and the input symbol is A, so current state moves right to state L (left). The green arrow labeled A:X, instructs the read/write head to write symbol X over A then moves to the left one position (new current state is L 'left'), as illustrated in the following diagram.



...	#	#	#	#	X	A	A	A	A	A	#	...

The green arrow now labeled #:1, the Turing machine movesback (left) to state R, and the read/write head writes symbol 1 over current symbol #, then move the head to the right one position (back to symbol X), since the current state is R (right), as in the diagram below:



...	#	#	#	1	X	A	A	A	A	A	#	...

Each step, the Turing machine will replace symbol A by X, increments binary numbers 0s to 1s and 1s to 0s, replaces Xs by #s then halts the execution, as the following diagram.

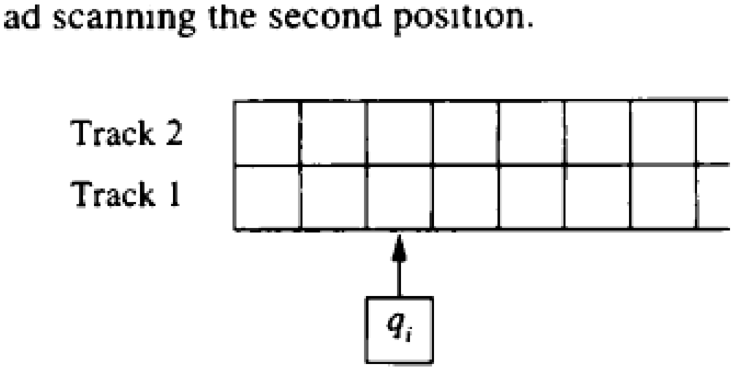
...	#	1	1	0	X	X	X	X	X	X	#	...
...	#	1	1	0	X	X	X	X	X	X	#	...
...	#	1	1	0	X	X	X	X	X	X	#	...
...	#	1	1	0	X	X	X	X	X	X	#	...
...	#	1	1	0	X	X	X	X	X	X	#	...
...	#	1	1	0	X	X	X	X	X	X	#	...
...	#	1	1	0	X	X	X	X	X	X	#	...
...	#	1	1	0	X	X	X	X	X	X	#	...
...	#	1	1	0	X	X	X	X	X	#	#	...
...	#	1	1	0	X	X	X	X	#	#	#	...
...	#	1	1	0	X	X	X	#	#	#	#	...
...	#	1	1	0	X	X	#	#	#	#	#	...
...	#	1	1	0	X	#	#	#	#	#	#	...
...	#	1	1	0	#	#	#	#	#	#	#	...

[4]

IV. Variants Of Turing Machine

The Turing machine had many variants, each variation supposed to increase the computing (processing) capability of the machine. And they must be equivalent to the standard Turing Machine.

Multi-Track Machine



A Multitrack tape machine has the tape divided into tracks. A tape position in an n-track contains n symbols from the tape limited alphabet. The diagram above shows a two-track tape with the read/write head scanning the second position. A position in this two-track tape machine is represented by the ordered pair [x, y], x is the symbol in track 1, y is the symbol in track 2.

Like the standard Turing machine, the two-track tape machine is a six-tuples:

$M = \langle Q, \Sigma, \Gamma, \delta, q_0, F \rangle$, where

- Q is a finite set of states;
- $\Sigma \subseteq \Gamma \setminus \{b\}$ is a finite set of *input symbols*, that is, the set of symbols allowed to appear in the initial tape contents;
- Γ is a finite set of *tape alphabet symbols*;
- $q_0 \in Q$ is the *initial state*;
- $F \subseteq Q$ is the set of *final* or *accepting states*;

With the finite set of states (*Q*), finite set of input symbols ($\Sigma \subseteq \Gamma$), set of limited tape alphabet (Γ), initial state ($q_0 \in Q$), and final state ($F \subseteq Q$)are the same as Turing machine. The transition functions δ of the two-track machine is defined as:

P$\delta(q_i, [x, y]) = [q_j, [z, w], d]$, where $d \in [L, R]$ [5]

Symmetric Turing Machine

The symmetric Turing machine is defined as a variant of Turing machine with a set of transitions (transition functions) of the form (*p, ab, D, cd, q*), where *p, q* are states, *ab*, *cd* are pairs of symbols and *D* is the moving direction of the read/write head. Example, if the direction *D* is left, the head is in state *p* above the tape symbol *b* with symbol *a* preceded it, then the head can move left, the state changes to *q* and the symbols *ab*replaced by *cd*. The opposite transition function (*q, cd, -D, ab, p*) can always be applied. If the moving direction *D* is right, the transition is analogues. The symmetric Turing machine looks at two symbols at a time and makes change to both.

Symmetric Turing machine was first introduced in 1982 by Harry R. Lewis and Christos apadimitriou when solving the problem, whether is there a path between two vertices s and t in an undirected graph? [6]

Multi-Dimension Turing Machine

This type of machine has multi-dimensional tape. The standard Turing machine could be seen as a two-dimensional machine with the head can move left or right and read or write on an unlimited tape. With a multi-dimensional tape like the checkerboard, the possible positions for the head to move is {N (north), E (east), S (south), W(west)} for a four-dimensional machine, or more such as {N, E, S, W, NW (northwest), NE (northeast), SE (southeast), SW (southwest)} [7]

V. Conclusion

The Turing machine was first introduced by Alan Turing in the 1930s, it was the simplest abstract machine for computing, but it still served as the model for the design of real world computers. Like modern computers, Turing machine has the same power in language recognitions, but also has restricts and limitations that can be used to explain some problems in the theories of computing such as: undecidability, complexity, computability, and halting problems.

Even as an abstract computing machine, the Turing machine can do most of the tasks as the real common computers that we are using daily. Their designed models might appear different, but they can accept the same languages – the recursive, enumerable languages. [8]

Turing machine was one of the first computing models that had some practical importance then eventually became the most established and standard model of computation.

1. Simplicity. The Turing machine is “simple” has a finite set of states, a finite set of alphabet where the input symbols come from, and the next state of the machine is depended on the current state and the input symbol, and the movement of the head.
2. Efficiency. Can do the tasks as required, this is a very important characteristic of the Turing machine.

The value of a theoretical concept (Turing machine)has had for practice is measured by considering all descendants that follow up the work, the results, and new ideas made came out from that theory concept. We can say the concepts of Turing machine have changed the world we live today. [9]

References:

[1] Michael Sipser, Mit, “Introduction To The Theory Of Computation”, 1997, Pws Publishing Company, Boston, Ma. 02116-4324. Pages: 1-3, 125-126

[2] From The Internet: Https://Www.Bing.Com/Search?Q=Application+Of+Turing+Machine&Qs=Nm&Pq=Turing+Machine+Applications&Sk=Hs1&Sc=8-27&Cvid=7b091805493f49469bccd054508a1ed8&Form=Qbre&Sp=2&Lq=0 Downloaded 1/3/2024 At 11:53am

[3] From The Internet: Https://En.Wikipedia.Org/Wiki/Turing_Machine Downloaded 1/1/2024 At 1:38pm Pages: 1-3

[4] From The Internet: Https://Introcs.Cs.Princeton.Edu/Java/52turing/ Downloaded 1/1/2024 At 12:49pm Pages: 1-5

[5] Thomas A. Sudkamp (2006). “Languages And Machines, 3rd Edition”.Addison-Wesley. Isbn 0-321-32221-5.Chapter 8.4: Multitape Machines, Pages: 263–264

[6] From The Internet: Https://En.Wikipedia.Org/Wiki/Symmetric_Turing_Machine Downloaded 1/6/2024 At 3:25 Pm

[7] Xavier, Eugene S.P. “Theory Of Automata, Formal Languages And Computation.”2005, New Age International (P) Limited, Publisher, New Delhi, India. Isbn(10): 81-224-2334-5, Isbn (13): 978-81-224-2334-1, Pages: 196

[8] Hopcroft, Motwani, Ullman, “Introduction To Automata Theories, Languages, And Computation 2e” 2001, Addison-Wesley, Ma., Isbn: 0-201-44124-1, Pages: 345, 355

[9] From The Internet: Https://Cs.Stackexchange.Com/Questions/9341/Practical-Importance-Of-Turing-Machines Downloaded 1/7/2024 At 3:25pm