ISSN: 2278 – 1323
International Journal of Advanced Research in Computer Engineering & Technology (IJARCET)
Volume 2, Issue 6, June 2013

Comparative Study of Two-way Finite Automata and Turing machine

Sumaiya Faizyab

Dept. of Computer Science and Engineering
Integral University
Lucknow, India

Abstract— Two-way finite Automata are termed as read only Turing machine. Two way finite automata is one of variant of Turing machine, which is blessed with an infinite tape, and could be used by tape head to read or write into its cell. Though Two way finite automata has it reservation as it is only read only with finite tape but many times unlimited storage is not required. This paper do comparative study to find out how Two way finite automata and Turing machine are different and whether any of the two have advantage over other or not.

Keywords—Turing machine, Two-way Finite Automata, Chomsky

I. INTRODUCTION

Turing machines, first described by Alan Turing (Turing 1937), as a simple abstract computational devices[1] intended to help investigate the extent and limitations of what can be computed. Turing proposed a class of devices that came to be known as Turing machines. In Alan Turing's 1936 paper on Computable Number with an application to the EntscheidungsProblem, he proposed a computing machine. Turing describes a machine that has an infinitely long tape upon which it writes reads and alters symbols. He further shows that a machine with the correct minimal set of operations can calculate anything that is computable, no matter the complexity.

Two way finite automata was explicitly introduced by Rabin and Scott, not ready to confined to a strict forward motion across Turing tape , they propose Two Way Finite automata in their famous paper Finite Automata and Decision Problem .They bring forth a two way finite automata over a finite alphabet as a system Hu=(S,M,s0,F) where S is a finite non-empty set (the set of alphabet ∑) the internal state of M is a function from S×∑ into L×S(the table of moves of Hu) s_0 is the element of S(initial state of Hu) and F is a subset of S(the set of designated final state of Hu)[2]

II. BASIC CONFIGURATION

Formally, a 2DFA is an octuple M = (Q, ∑, ├, a, ┤, s, t, r), where Q is a finite set (the states), ∑ is a finite set (the input alphabet), ├ is the left end marker, ┤ is the right end marker, ϙ : Q× (∑U{ ├,┤ }) → (Q × {L,R}) is the transition function (L,Rstand for left and right, respectively),s ε Q is the start state, t ε Q is the accept state. Two way finite automata accept a language by fixing an input in between two end marker and if after some finite moves it reaches to final state it is considered to be accepted. The configuration of two way automata is supposing a string x belong to Language L is need to checked then fix an input x ε ∑*, say x = a1a2 …..a_n. Let a_0 = ├ and a_{n+1} = ┤. Then $a_0a_1a_2$ …. a_na_{n+1} becomes ├ x ┤.

A configuration of the machine on input x is a pair (q, i) such that q ε Q and 0 ≤ i ≤n + 1. Informally, the pair (q, i) gives a current state and current position of the read head. The start configuration is (s, 0), meaning that the machine is in its start state *s* and scanning the left end marker. At any point in time, the machine is in some state q with its read head scanning some tape cell containing an input symbol or one of the end markers. Based on its current state and the symbol occupying the tape cell it is currently scanning, it moves its read head either left or right one cell and enters a new state. It accepts by entering a special accept state t and rejects by entering a special reject state r. The machine's action on a particular state and symbol is determined by a transition function that is part of the specification of the machine.

Turing Machine like finite automata does contain finite control and tape but similarities end over here there is much more to it. Turing machine has tape but infinite, most of the time left end is closed and its right end has long infinite tape. Unlike finite machine it has read/write head which could move left/right or stay at the same place. Formally Turing machine is 5 tuple T = < Q, Σ, Γ, q_0, δ >where Q is a finite set of states, which is assumed not to contain the symbol h. The symbol h is used to denote the halt state, machine comes to halt only when it accept the input unlike two way finite automata which has two separate *accept* and *reject* state. Σ is a finite set of symbols and it is the input alphabet. Γ is a finite set of symbols containing Σ as its subset and it is the set of tape symbols. q_0 is the initial state. δ is the transition function but its value may not be defined for certain points. It is a mapping from Q ×($\Gamma \cup \{ \Lambda \}$) to (Q $\cup$ { h }) ×($\Gamma \cup \{ \Delta \}$) ×{ R , L , S } .

III.LANGUAGE ACCEPTOR

Rabin and Scott, Shepherdson [3], bring forth that both in the deterministic and in the nondeterministic versions of two way finite automata, have the same computational power of one-way automata, namely, they characterize the class of regular languages.

Chomsky hierarchy [4] is a containment hierarchy of classes of formal grammar. Chomsky hierarchy consist of four level of formal languages Type-0 or recursively enumerable language, type-1 or context-sensitive grammar, type-2 grammar or context free grammar and type-3 or regular

language. Every regular language is context-free, every context-free language, not containing the empty string, is context-sensitive and every context-sensitive language is recursive and every recursive language is recursively enumerable. One of the theorems says about regular language and recursive language

Theorem: $L_{Reg} \rightarrow L_{Rec}$[5]

Proof: Recursive Language lie beneath phase structure and recursive enumerable languages in Chomsky hierarchy of language and they are decidable language, is well established fact, if we consider a palindrome it is decidable thus could be included in recursive language but are not recognized by FSA thus is not a regular language .This reestablished fact that $L_{Reg} \in L_{Rec.}$

This leads us to conclude that regular languages are included as part of the recursive languages. Turing Machine can easily act as an FSA all transition will be of form

$$\delta (q1, a) = (q2, R)$$

Thus it could be concluded that Turing Machine can accept regular language, whereby we only read from the tape and transition internally from one state to the other, never writing on the tape or moving in the other direction or if one is to say more specifically then it is recognized by a read only Turing machine ,that use only a constant amount of space on their work tape, since any constant amount of information can be incorporated into the finite control state via a product construction (a state for each combination of work tape state and control state).

Turing machine recognize recursively enumerable language.

IV. PROBLEM

Now we will do a comprehensive study with a problem and see how it is accepted by Turing Machine and Two way finite automata. We are going to take a simple problem and analyses the difference in accepting of L= $\{a^n b^m | m, n>0\}$[6].

The two-way finite automata read a magnetic tape, move one block in either side left/right at each move as it is two way finite automata. The following 2 state of 2FA over $\sum = \{0, 1\}$ that accept tapes containing n number of a's with m number of b's enclosed by left and right end marker. The 2FA has starting state q0, q0q1as its final state and r reject state .The formal description of two-way F.A is

Q = {q0, q1}

$\sum$ = {a, b}

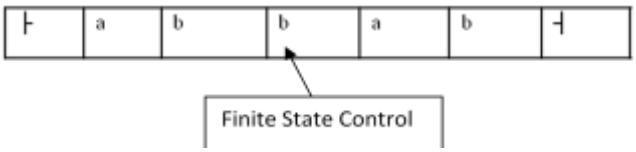
s=q0

t=q1

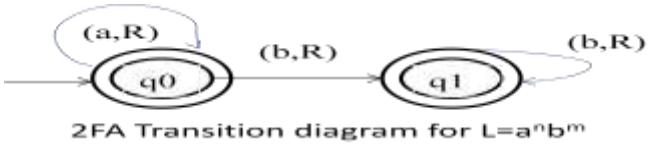
The transition function is given by the following table:

States	├	a	b	┤
qo	(q0,R)	(q0,R)	(q1,R)	(r,L)
q1	-	(r,L)	(q1,R)	(q1,L)

The machine starts with its start state q0 with its read head pointing to the left end marker. The transition $\delta (q_i, a) = (q_j, R)$ shows that there is a transition from state q_i to q_j on reading input alphabet a and machine takes a right move. Lets us suppose the input string is abbab the first two symbol is already scanned and the current input is a the machine is in the state q_i the transition is shown



2FA start with start state q0 and read left end marker ├ and move to right to come to the input alphabet, if the string start with a then finite state move to right direction δ (q0, b) if q0 state encounter with b then state change to q1 which also read all the b's in the string. The thing important to note is that if q0 meet right end marker ┤then it enter into reject state and its head moves to left direction so that it is not struck. Similarly when state q1 reaches to right end marker ┤it comes to it accept state that is q1 and head move to left direction and stays.



2FA Transition diagram for $L=a^n b^m$

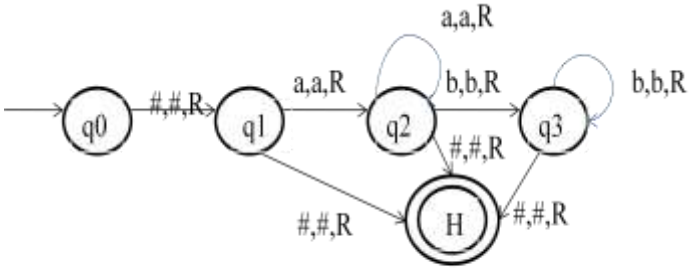
Turing machine read the input string from one state to another .When it reach to the end of the input string, the machine halts when the read /write head read # (the blank) in the cell. The Turing machine for language L= $\{a^n b^m | m, n>0\}$ is possible because regular language are acceptable by read only Turing machine but difference lie with the basic structure of Turing machine, the string is put on infinite tape, the number of state consumed in accepting the language, acceptability will also be determine if it each to halt state otherwise it will be struck forever. Another difference lie with Turing machine, it contain blank symbols which is missing in two way finite automata .The formal description of Turing machine will be Q = {q0, q1, q2, q3}

$\sum$ = {a, b}

Γ = {a, b, #}

The transition function will be

states	a	b	#
q0	-	-	(#,#,R)
q1	(a,a,R)	-	(#,#,R)
q2	(a,a,R)	(b,b,R)	(#,#,R)
q3	-	(b,b,R)	(#,#,R)
h	-	-	(#,#,h)



Transition Diagram for $L=a^n b^m$ accepted by Turing machine

As transition table and diagram shows input start with # to mark the left end of string, TM read # and move to right to get across with input alphabet, if the string contains only # means empty then it will move to halt state and accept the string. If on state q1 it read alphabet as many time as in string then it will move to right and change its state to q2, on reading alphabet b as many times in the string it will transit to state q3 comes to halt.

V. COMPLEXITY

Now we will tried to find out the complexity of Turing machine and Two-way finite in accepting the language L= $\{a^n b^m\}$ .Though we add very little in this regard but it may lead to some meaningful conclusion. Complexity could be in term of space and time .Time complexity is concern with number of moves and space with storage .But we will stick to time complexity because of Turing machine. Time complexity of TM is how many times the tape moves when the machine is started on some input. Space complexity refers to how many cell of the tape are written to when the machine run. Since, in this case, read only Turing machine is employed we will focus only on time complexity.

The complexity of Two-way finite automata for L= ($\{a^n b^m\}$ is based on the maximum number of moves made by the machine.Machine start on the left most symbol, it always move right, at each step (i) read the number for a's (ii) for each pair of a's read and mark number of b's.Thus at each phase the maximum number of move (approximately) will be the halves (n/2) of the input string. On reaching at right end marker 2FA accept the string.At each individual step O (n) moves. Thus notably the complexity of Two-way finite automata in recognizing L= $\{a^n b^m\}$ will be polynomial time i.e. O (n^2).

As for Turing machine, each moves read to see if # for unmarked a's is equal to # of unmarked b's.Read again accepting every other a's starting with the first and every other b's starting with first. If # remains on the tape accept it else reject.So number of phases is O ($\log_2 n$).Time for each individual phase will be O (n).

So the number of step by the Turing machine is at most O (nlogn).

ACKNOWLEDGMENT

It brings me immense pleasure to present this paper, for which I need to thanks to the people who make it possible. I would love to thank my colleague Ms. Gargi Chaterjee who motivates me to write this piece of work and become my source of inspiration. My sincere thanks to another very senior mentor Mr. Kuldeep Singh who provide me help and patiently listen my problems.

REFERENCES

[1] A.M TURING "On Computable Number with an application to the ENTSCHEIDUNGSPROBLEM "
[2] M.O Rabin and D.Scott " Finite Automata and Their Decision Problems"
[3] J. *C.* Shepherdson, "The reduction of two-way automata to one-way automata," ***IBM Journal,*** **3,** 198-200 (1959).
[4] Noam Chomsky"On Certain Formal Properties of Grammar" *Information and Control9. 137-167 (1959)*
[5] http://staff.um.edu.mt/afra1/teaching/coco5.pdf
[6] Rajendra Kumar - "Theory of Automata, Lang & Computation"