Deque Automata for all classes of Formal languages

B. Asha latha[1]
Department of computers
SRKIT Engineering
Vijayawada Andhra Pradesh (India)

T.Vishnupriya[2]
Department of Electronics
SRKIT Vijayawada, Andhra Pradesh (India)

N.Himabindu[3]
Department of computers
KBN College of Vijayawada, Andhra Pradesh (India)

Abstract: The purpose of computation involves solving problems by communicating them to a computational model by means of a suitable language .A number of languages have been developed for this purpose. To recognize these languages some computational models has been developed and they are finite state machine, push down automata, queue automata and turing machines. But these machines are restricted to only one specific formal languages like regular, context free ,etc. In this paper we proposed a machine called a Dequeue automaton that is capable of recognizing different classes of automata. We also shown that the simulation results from the Deque automata.

Keywords: Formal languages, Finite automata, PDA ,TM.

I. Introduction

A finite automaton was the first abstract model as well as the mathematical model of digital computers. It is very powerful model of computation. It can recognize and accept regular languages. But finite automata have limited memory(states) which prevents them accepting Context free languages .Since memory is a limitation of finite automata ,a memory element is added as a stack, in order to made finite automata a powerful machine and to accept Context free languages. That new type of computational model is known as a Push down automata.PDA is similar to finite automata except that it has an extra memory unit stack. Stack is defined as a data structure where insertion and deletion of any element is possible only at one end called top of the stack.[1].

The automata with queue memory was constructed in a similar way as the PDA, however the new type of memory of QA is queue. The definition of queue automata is similar to that of PDA. The difference concerns the type of memory. The main advantage of QA is it is equivalent to Turing machine. That is a TM can be simulated by a QA that keep a copy of the TM‘s contents in its queue at all times with two special marks. One for the end of TM’s head position and one for the end of the tape. Its transitions simulate those of the TM by running through the whole queue, popping off each of its symbols and re-enqueing either the popped symbol or near the head position. A queue machine can be simulated by a TM but more easily by a multi tape TM which is known to be equivalent to a normal single-tape machine.

But the PDA is not able to recognize the Context sensitive languages and Recursively Enumerable languages. To recognize the Context sensitive languages and Recursively Enumerable languages another automaton that is Turing machine was developed. The following table summarizes each class of the formal language and its corresponding automaton that recognizes it.

II. Categories of languages and Automaton

S.NO	Formal language	Automaton
1	Regular language	Finite automaton
2	Context free language	Non deterministic Push down automaton
3	Context sensitive and Recursively enumerable languages	Turing machine
4	Context free languages	Queue Automaton

Table: 1 Different types of Automaton

Every regular language is Context free, every context free language, not containing empty string is context sensitive and every recursive language is recursively enumerable. These are all proper inclusions, meaning that there exists recursively enumerable languages which are not context sensitive ,context sensitive languages which are not context free and context free languages which are not regular.

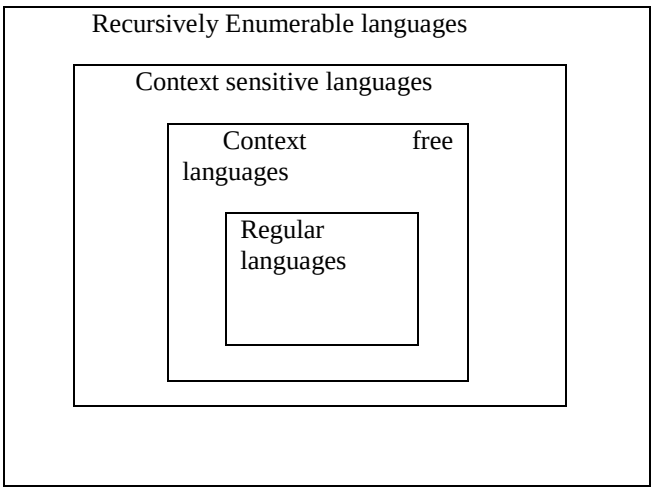

Fig : 1 Categories of languages

II. Deque computational model

The Deque Automaton is an extension of Queue automaton .It uses double ended queue as a memory element. The Dequeue is auxiliary storage element ,that contains list of items from a finite alphabet 'ζ' where the alphabets can be read ,inserted, or deleted from the both ends of the queue. That is we can perform the three operations namely reading, inserting, deleting from the front end as well as rear end of the deque. The main advantage of the Dequeue automaton is all the formal languages can be recognized by using it.

III Defining the automata

A deque automata is a hex-tuple machine. we can define it mathematically as

$$M=(Q,\sum, \zeta,\delta,S,F)$$

Where Q=non empty set of finite state (s)

$\sum$=non empty set of input alphabet

ζ=deque alphabet

(δ)=transision function

S=Starting state

F=Final state

III. Operations of Deque

Consider the expression $(p,a,\alpha \ldots.\beta), (q,\gamma\ldots\theta) \varepsilon \delta$,where $\{p,q\} \varepsilon Q$

$a\varepsilon \sum$

$\alpha,\beta \varepsilon \zeta$

We can describe the above expression as if the automata are in the state

'p' with' α' as the front element and' β' as the back element of the Deque and reading the current input symbol from the tape as 'a' ,then the machine enters into a state 'q' by changing the alphabets of the que as α=γ or β=θ .If the machine is non deterministic then there is possibility that a=e (empty string).There is always a chance that p=q [2] There are six basic operations involved with the deque.

1.PUSH R: Pushing the input alphabet from the right of the deque

Examples:
((p,a,e) ,(q,a)) <= PUSH R means pushing the symbol 'a' on an empty deque from its right.

((p,a,a),(q,aa)) <= PUSH R means pushing the symbol 'a' on' a' into the deque from its right

((p,a,b),(q,ba)) <= PUSH R means pushing the symbol 'a' on' b' into the queue from its right

((p,b,a),(q,ab)) <= PUSH R means pushing the symbol 'b' on 'a' into the deque from its right

((p,b,b),(q,bb)) <= PUSH R means pushing the symbol 'b' on' b' into the deque from its right

Where {p,q}εQ

{a,b}ε∑

2.PUSH L: Pushing the input alphabet from the left of the deque

Examples:
((p,a,e),(q,a)) <=PUSH L means, pushing the symbol 'a' on an empty deque from its left

((p,a,a),(q,aa)) <=PUSH L means pushing the symbol 'a' on' a' into the deque from its left

((p,a,b),(q,ab)) <=PUSH L means pushing the symbol 'a' on' b' into the deque from its left

((p,b,a),(q,ba)) <=PUSH L means pushing the symbol 'b' on' a' into the deque from its left

((p,b,b),(q,bb)) <=PUSH L means pushing the symbol 'b' on' b' into the deque from its left

Where {p,q}εQ

{a,b}ε∑

3.POP R: removing or deleting an item from the right of the deque

Examples:
((p,u,a),(q,e)) <= POP R , delete the symbol 'a' from right of the deque

((p,u,ab),(q,a)) <= POP R , delete the symbol 'b' from right of the deque

((p,u,ba),(q,b)) <= POP R , delete the symbol 'a' from right of the deque

((p,u,b),(q,e)) <= POP R , delete the symbol 'b' from right of the deque

Where {p,q}εQ

u ε∑*

{a,b}εζ

4.POP L : Removing or deleting an item from the left of the queue

Examples:
((p,u,a),(q,e)) <= POP L ,means delete the symbol 'a' from the left of the deque

((p,u,ab),(q,b)) <= POP L means delete the symbol 'a' from the left of the deque

((p,u,ba),(q,a)) <= POP L, means delete the symbol 'b' from the left of the deque

((p,u,b),(q,e)) <= POP L, means delete the symbol 'b' from the left of the deque

((p,u,e),(q,e)) <= POP L, means no symbol is present deque to delete and 'q' is the halting state.

5.SENSE R : Read the character from the input tape on the right of the current position of read head.

6.SENSE L:Read the input symbol on the left of the current position of the reading head.

IV. Representation of the deque

We can represent the transitions of the deque by following the notation Q X ∑* X ζ* where the first component is the machine state, second component is the input symbols, and the third component is the alphabets of deque reading from left to right. For example

Consider the notation (p,abc,ABC) 'p' is the present state 'abc' is the input string to be read,'ABC' is the content of the deque read from front to rear.

V. Deriving the other models from Deque

1. Queue automata from Deque: If we want to perform queue automata on deque we have to made some assumptions and these assumptions are restrict the use of deque operations only to PUSH L,POP R, and SENSE R.

Example: $L=a^nb^n$ (n>0) the instantaneous descriptions are as follows

$((q_0,a,e),(q_1,a))$ <=SENSE R,PUSH L

$((q_1,a,a),(q_1,a))$ <=SENSE R,PUSH L

$((q_1,b,a),(q_2,e))$ <=POP R

$((q_2,u,e),(q_3,e))$ <=POP R

Where $Q=\{q_0, q_1, q_2, q_3\}$

$\sum=\{a,b\}$

$\zeta=\{a\}$

$S=q_0$

$F=q_3$

2. Push down automata from deque: If the deque is compelled to use only PUSH L, POP L, SENSE R then the deque can be treated as PDA. Let take the same example $L=a^nb^n$(n>0) The instantaneous descriptions are as follows

$((q_0,a,e),(q_1,a))$ <= SENSE R,PUSH L

$((q_1,a,a),(q_1,a))$ <= SENSE R,PUSH L

$((q_1,b,a),(q_2,e))$ <= POP L

$((q_2,u,e),(q_3,e))$ <= POPL

Where $Q=\{q_0, q_1, q_2, q_3\}$

$\sum=\{a,b\}$

$\zeta=\{a\}$

$S=q_0$

$F=q_3$

3. Finite automata from the Deque automata: In finite automata there are two cases, the first case is DFA and the second Case is NFA. Every finite automata can be viewed as a Deque automata having no operation on the deque.

Let $M=(Q,\sum,\zeta,\delta,q_0,F)$ be a DFA

$M^L=(Q,\sum,\zeta,\delta^L,q_0,F)$ be aDeque automata

Case :1 To accept languages accepted by the DFA, the transition function δ^1 is defined as

$$\delta^1 = \{ ((p,u,e),(q,e)) : (p,u,q) \varepsilon \delta \}$$

Example : consider the regular language that accept three successive zeros

The transitions are $(\delta) : (q_0, 1) = (q_0)$

$(q_0,0) = (q_1)$

$(q_1,0) = (q_2)$

$(q_2,0) = (q_3)$ Here q_3 is the final state.

Case : 2 Any NFA can be converted to an equivalent DFA and the DFA is derived from the Deque automata.

4. Turing machine from the Deque automata : To derive Turing machine from the deque we need to perform all operations of Deque .Let us consider an example L= $a^n b^n c^n$,and the

instantaneous descriptions are as follows

$((q_0,a,e),(q_1,a))$ <=SENSE R,PUSSH L

$((q_1,a,a),(q_1,a))$ <=SENSE R,PUSH L

$((q_1,b,a),(q_2,b))$ <= SENSE R ,PUSH R

$((q_2,b ,b),(q_2, b))$ <= SENSE R,PUSH R

$((q_2,c,a\&b),(q_3,e))$ <= POP L and POP R

$((q_3,u ,be (q_4, b))$ <= POP L and POP R here q_4 is the final state

VI.Conclusions

In this paper from deque automata how remaining automatas are derived is described. The simulation results for all formal languages are also shown.

VII. Future scope

Researches are going on with deque of deque to find the intersection between the deque automata

References

[1]. *Introduction to automata theory languages and computation by ULLAMAN*
[2]. Bhattacharjee, A.,and Debnath, B.K.,"Queue Automata ".