Analysis of Different Methods to Solve a Problem $L = a^n b^{2n}$ for all $n >= 1$ using a Turing Machine in Terms of their Complexity

Abhishek Shukla
MANIT Bhopal
India
abhishekshukla94@gmail.com

ABSTRACT: *This paper is an attempt to compare different methods to solve some context free as well as context sensitive language [3] using Turing machine. Here we compare complexity [9] of two different methods to solve language like* $L= a^n b^{2n}$ *for all n>=1.This paper give an idea to the reader to develop and analyse different types of methods to solve computational problem in efficient way.*

Keywords: Context Free Languages [1], Complexity, Turing machine [2]

Received: 17 April 2018, Revised 21 May 2018, Accepted 29 May 2018

DOI: 10.6025/tmd/2018/6/2/43-46

1. Introduction

We have studied about Turing machine i.e. it is a tape which has infinite capacity to store data within the tape. It usages special character "#" used for blank within the tape. It has a read/write head which can move either Left or Right and can read or write within the tape one character at a time. A Turing machine is used to accept for those languages which are not accepted by DFA or PDA. Turing machine is an acceptor for the languages those are generated by unrestricted grammar [7]. A Turing machine is used to accept context sensitive, recursive [5], recursively enumerable languages [4].

Mathematically a Turing machine Tm can be described as follows.

$$Tm = \{Q, \Sigma, \tau, q_0, \delta, F, \#\}$$

Where

Q = {A finite Set of all internal states}

Σ = {A finite Set of input alphabets}

τ = {Finite set of symbols called the tape alphabet}

q_0 = {initial state}

δ = Is a transition function defined as follows

$$\delta : Q \times \tau \rightarrow Q \times \tau \times \{L, R\}$$

Here δ is a partial function on $Q \times \tau$.

F = {A finite set of final states}

= # ε τ; Is a special symbol used for blank on the tape

Now we will described bellow two different methods to compute the language $L = \{a^n b^{2n}; \forall\, n >= 1\}$.

Method 1(Overwriting Method):

A *Tm* for the language $L = \{a^n b^{2n}; \forall\, n >= 1\}$ is defined by

$Tm_1 = \{Q, \Sigma, \tau, q_0, \delta, F, \#\}$

Where

$Q = \{q_0, q_1, q_2, q_3, q_4, q_5\}$

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

$\tau = \{a, b, x, y, z, \#\}$

$q_0 = \{q_0\}$

$F = \{q5\}$

And its transition function δ is defined as follows

$\delta(q_0, a) = (q_1, x, R)$

$\delta(q_1, a) = (q_1, a, R)$

$\delta(q_1, y) = (q_1, y, R)$

$\delta(q_1, b) = (q_2, y, L)$

$\delta(q_2, y) = (q_2, y, L)$

$\delta(q_2, a) = (q_2, a, L)$

$\delta(q_2, x) = (q_0, x, R)$

$\delta(q_0, y) = (q_3, y, R)$

$\delta(q_3, y) = (q_3, y, R)$

$\delta(q_3, z) = (q_3, z, R)$

$\delta(q_3, b) = (q_4, y, L)$

$\delta(q_4, y) = (q_4, y, L)$

$\delta(q_4, z) = (q_4, z, L)$

$\delta(q_4, x) = (q_3, z, R)$

$\delta(q_3, \#) = (q_5, \#, L)$

Method 2 (Non Overwriting Method):

A *Tm* for the same language $L = \{a^n b^{2n}; \forall\, n >= 1\}$ is defined by

$$Tm_2 = \{Q, \Sigma, \tau, q_0, \delta, F, \#\}$$

where

$Q = \{s_0, s_1, s_2, s_3, s_4, s_5\}$

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

$\tau = \{a, b, x, y, \#\}$

$q_0 = \{s_0\}$

$F = \{s_5\}$

and its transition function δ is defined as follows

$\delta(s_0, a) = (s_1, x, R)$

$\delta(s_1, a) = (s_1, a, R)$

$\delta(s_1, y) = (s_1, y, R)$

$\delta(s_1, b) = (s_2, y, R)$

$\delta(s_2, b) = (s_3, y, L)$

$\delta(s_3, y) = (s_3, y, L)$

$\delta(s_3, a) = (s_3, a, L)$

$\delta(s_3, x) = (s_0, x, R)$

$\delta(s_0, y) = (s_4, y, R)$

$\delta(s_4, y) = (s_4, y, R)$

$\delta(s_4, \#) = (s_5, \#, L)$

Similarly for $a^n b^{3n}; \forall\, n >= 1$ *Method1* would become little bit complicated and take more steps but *Method2* will take just one more step than $a^n b^{2n}$ and for $a^n b^{4n}$ *Method1* goes more complicated and *Method1* will take one more step than $a^n b^{3n}$ and so on. As *b*'s integral multiple of *n* increases, number of steps as well as number of states in *Method2* increases linearly and in *Method1* it increases randomly.

A table for *Method2* is given as follows.

2. Conclusion

From the above two methods we analyze that *method2* is efficient than method1 and take less number of steps to perform computation hence the time complexity of *method2* is fewer than *method1*.Though the space complexity of the above two methods will be same as they both have the same number of states. At designer point of view also is easier than *method1*. In the similar way we can see that for the language $L = \{a^n b^n c^{2n} \forall\, n >= 1\}$ which is not be accepted by either DFA or PDA but turing

Language	No. of steps	No. of States
$a^n b^{3n}$	11	6
$a^n b^{3n}$	12	7
$a^n b^{3n}$	13	8
$a^n b^{3n}$	14	9
$a^n b^{3n}$	15	10

Table 1

machine , method2 will be efficient to method1. The above two comparison gives an idea to the readers to design some more algorithms which would be easier in computational nature and perform in fewer time.

References

[1] Linz, Peter. Narosa Publication, An intoduction to Formal Languages and Automata fourth Edition.

[2] Alfred, V., Aho, Monica, S., Lam, Sethi, Ravi., Jeffery, D. Ullman by Pearson Education, Inc. and Dorling Kindersley Publication, Compilers Principles, Techniques and Tools Second edition. Language No. of steps No. of States

[3] Introduction to automata theory, Languages and computation/John E. Hopcroft, Rajeev Motwani, Jeffery D. Ullman-2nd edition.

[4] http://www-db.stanford.edu/ullman/ialc.html

[5] Introduction to Languages and The Theory of Computation by John C. Martin, 2nd EDN) ISBN-0-07-463722-3.

[6] Theory of computation by Bernard M. Moret Pearson –ISBN-10:8131708705.

[7] Fundamentals of Theory of computation by Raymond Greenlaw & H. James Hoover (Harcount) ISBN-81-7867-036-4

[8] Elements of Theory of computation by Lewis & Papadimitrou, PHI ISBN- 81-203-1271-6

[9] Elements of Descrete Maths By C. L. LIU TMH 2nd EDN ISBN-0-07-043476-X