IJIRSET
International Journal of Innovative Research in
SCIENCE | ENGINEERING | TECHNOLOGY

e-ISSN: 2319-8753 | p-ISSN: 2347-6710

INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH

IN SCIENCE | ENGINEERING | TECHNOLOGY

Volume 10, Issue 8, August 2021

INTERNATIONAL STANDARD SERIAL NUMBER INDIA

Impact Factor: 7.569

9940 572 462 | 6381 907 438 | ijirset@gmail.com | www.ijirset.com

| e-ISSN: 2319-8753, p-ISSN: 2347-6710| www.ijirset.com | Impact Factor: 7.569|

|| Volume 10, Issue 8, August 2021 ||

| DOI:10.15680/IJIRSET.2021.1008104 |

A Newly Proposed Empirical Composite approach based Ingenious Mutative Regulative Automata (MRA) for CSL and CFL

Shivankur Thapliyal [1] , Pooja Bisht[2]

Assistant Professor, Department of Computer Science and Engineering, Doon Institute of Engineering and Technology, Rishikesh, Uttarakhand, India [1]

Assistant Professor, Department of Computer Science and Engineering, Doon Institute of Engineering and Technology, Rishikesh, Uttarakhand, India [2]

ABSTRACT: In the circumstances of Computational Complexity Theory ,To Validate any Language using various state machines with lower computational cost and to achieved greatest reliability with fastest response time with better throughput are one of the most significant challenging concernable issue from some recent periods of time. In this paper a novel state machine name says Mutative Regulative Automata (MRA) have been proposed. The basic purpose to designed this newly created significant state machine is that to achieve desired validation techniques with fastest response time with better reliability and as well as very lower computational cost. Here In this paper we proposed two methodologies to designed this MRA. Both methods are performs similar types of tasks, but the way of designing MRA using these methods are totally differ as relative to other methods. Generally Mutative Regulative Automata are responsible to validate Context Sensitive Language (CSL) and Context Free Language (CFL). For MRA no two separate state machines are required to validate CSL and CFL, a single machine of MRA are responsible to validate both CSL and CFL Language, that's why it's called Composite or Hybrid approach based state machine. A one main significant factor is that this MRA state machine doesn't contains any memory element to validate any CSL and CFL Language, that's why it's computational process becomes so fastest with reliability and also optimized computational cost.

KEYWORDS: Mutative Regulative Automata, Hybrid approach of MRA, CFL state machine, CSL state machine, MRA.

I. INTRODUCTION

Modelling some deterministic and non-deterministic State Machines (SM) [1] are one of the most significant and dominant challenging area for validate any Language to draw State Machine (SM) [2] model, but a one major issue is that when a language becomes more complex and typical then their respective State Machines (SM) [2] becomes more vigorous or mighty and hard to understand , and there are some various types of desired States Machines (SM) [2], such as: Turing Machine (TM) [3], Linear Bounded Automata (LBD) [3], Push Down Automata (PDA) [4], and Finite Automata (FA), but each machines are to be responsible for recognizing or to identify some specific languages, but here in this paper two methods come across which is an state of the art of two respective machines, which is Push Down Automata (PDA) [4] and Linear Bounded Automata (LBD) [3]. These two novel proposed machine model are similar in nature and to do similar types of tasks, but the way of working are totally differ than some other ones. These methods have some proficiency that it's a common machine for those languages, which are accept or validate by PDA [4] and LBA [3]. It's a fusion of these two machines (PDA [4], LBA [3]) with adopting a new methodology. Generally, the main locus which make PDA [4] totally differ than LBA is that the number of comparisons between them, if there exists only one comparison between them, then it's must excepted by PDA [4], and if more than one then it's must excepted by LBA [10], e.g. $L = \{a^n b^n\}$, it's a language which explicate that the number of a's and b's are equal and a's always placed to be first than b's, so the number of comparisons between a's and b's are only one, so this type of languages are easily accepted by Push Down Automata (PDA) [4], but languages such as e.g. $L = \{a^n b^n c^n\}$, which contains more than one comparisons such as, that the number of a's, b's and c's are to be equal and a's placed before b

| e-ISSN: 2319-8753, p-ISSN: 2347-6710| www.ijirset.com | Impact Factor: 7.569|

|| Volume 10, Issue 8, August 2021 ||

| DOI:10.15680/IJIRSET.2021.1008104 |

and b's before c's, so here the number of comparisons between more than one so this type of languages will easily accepted by Linear Bounded Automata (LBA) [3]. But the new proposed model are the mixture of these two machines respectively PDA and LBA. One more disadvantage to use these traditional machines such as PDA [4] and LBA [3] are the type of memory elements, such as PDA [4] used one Stack (LIFO) type of Data Structure for one comparison, while LBA [3] used two memory elements, which are two Stacks (LIFO) types of Data structure for two comparisons. But a new proposed model will not use any Stack type of Data Structure, its uses 8 bit Registers generally Accumulator (AC) for storing some intermediate results during computations. The well-known appropriate structure of these newly proposed machines are in the next section.

II. SOME PREVIOUSLY STATE MACHINES TO ACCEPT SUCH TYPE OF LANGUAGES E.G. $L = a^n b^n \; for \; n > 0$.

Generally Push Down Automata (PDA) [4] are used to identify Context Free Languages (CFL) [5]. CFL accepts those type of Languages, which contains some counting and sequencing patterns. Languages such as $L = a^n b^n \; for \; n > 0$ are used to identify through PDA [9]. DFA (Deterministic Finite Automata) [7] are used to identify RL (Regular Languages) [6], which contains some finite number of information, but PDA (Push Down Automata) [4], which are generally used to identify Context Free Languages (CFL) [5], which contains some infinite number of information. A well-defined PDA structure are as follows:

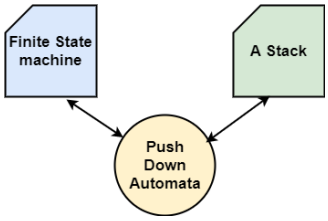

fig a: PDA Structure

A Pushdown Automata (PDA) contains three components –

- ✓ **an input tape,**
- ✓ **a control unit, and**
- ✓ **a stack with infinite size.**

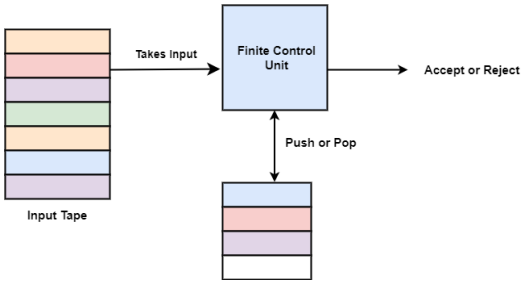

Fig b: components of PDA

The stack head trace the top symbol of the stack.
A stack perform following two major operations –

- ✓ **Push** – a new symbol is added at the top.
- ✓ **Pop** – the top symbol is read and removed.

A PDA may or may not read an input symbol, but it has to read the top of the stack in every transition.

| e-ISSN: 2319-8753, p-ISSN: 2347-6710| www.ijirset.com | Impact Factor: 7.569|

|| Volume 10, Issue 8, August 2021 ||

| DOI:10.15680/IJIRSET.2021.1008104 |

A PDA can be formally described as a 7-tuple (Q, ∑, S, δ, q_0, I, F) –

- ✓ **Q** is the finite number of states
- ✓ ∑ is input alphabet
- ✓ **S** is stack symbols
- ✓ **δ** is the transition function: Q × (∑ ∪ {ε}) × S × Q × S*
- ✓ **q_0** is the initial state (q_0 ∈ Q)
- ✓ **I** is the initial stack top symbol (I ∈ S)
- ✓ **F** is a set of accepting states (F ∈ Q)

PDA accept any input string using two methods:

- ➢ **Acceptance by Final State:** If any PDA reached any final state using zero or more moves after reading the entire input then those PDA adopted this methodology.
  Let P =(Q, ∑, Γ, δ, q0, Z, F) be a PDA. The language acceptable by the final state can be defined as:
  L(PDA) = {w | (q0, w, Z) ⊢* (p, ε, ε), q ∈ F}

- ➢ **Acceptance by Empty Stack:** On reading the input string from the initial configuration for some PDA, the stack of PDA gets empty.
  Let P =(Q, ∑, Γ, δ, q0, Z, F) be a PDA. The language acceptable by empty stack can be defined as:
  N(PDA) = {w | (q0, w, Z) ⊢* (p, ε, ε), q ∈ Q}

a. Equivalence of Acceptance by Final State and Empty Stack [8,11,12]

If L = N(P1) for some PDA P1, then there is a PDA P2 such that L = L(P2). That means the language accepted by empty stack PDA will also be accepted by final state PDA.
If there is a language L = L (P1) for some PDA P1 then there is a PDA P2 such that L = N(P2). That means language accepted by final state PDA is also acceptable by empty stack PDA.

b. An Illustrative Example of PDA

The diagrammatic representation of PDA are as follows, which accepted following types of Languages $= a^n b^n \; for \; n > 0$, if we take n=3 , then it contains following form:

$L = a^n b^n \; for \; n = 3,$
$L = a^3 b^3 \; | \; aaabbb \; for \; n = 3,$

Now how PDA works for that language, it's detailed diagrammatic representation are as follows:

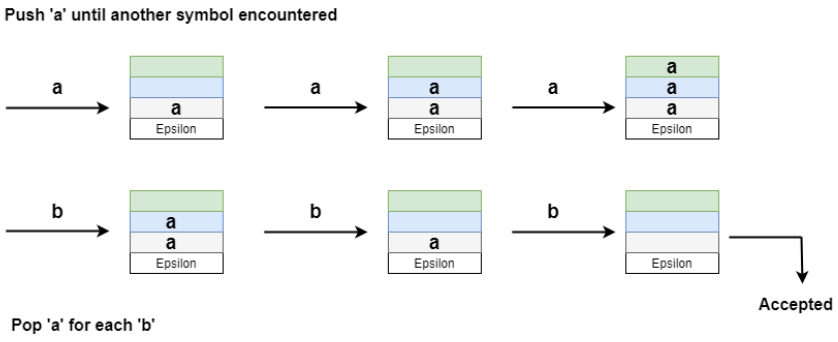

Fig c: PDA concept for Language $L = a^3 b^3 \; | \; aaabbb \; for \; n = 3$

| e-ISSN: 2319-8753, p-ISSN: 2347-6710| www.ijirset.com | Impact Factor: 7.569|

|| Volume 10, Issue 8, August 2021 ||

| DOI:10.15680/IJIRSET.2021.1008104 |

In this diagram firstly we push all a's in the stack, and after encountered b's, for each b, each a should be popped on to the stack, after finishing all b's, if the stack will be empty, then the input string have to be accepted.

The diagrammatic representation of these Language $\boldsymbol{L = a^n b^n\ for\ n > 0}$ PDA are as follows:

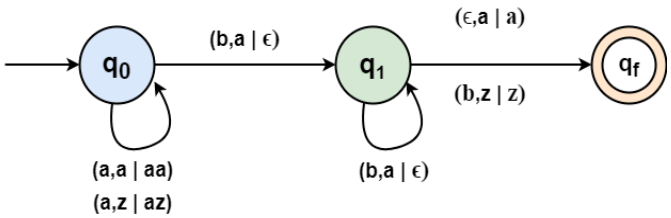

Fig d: PDA for Language $\boldsymbol{L = a^n b^n\ for\ n > 0}$

III. A NEW PROPOSED MUTATIVE REGULATIVE AUTOMATA (MRA) STATE MACHINE FOR CSL AND CFL

We designed Mutative Regulative Automata (MRA) by using two methods, each methods perform similar tasks, but the way of performing tasks are different.

First method of MRA:
In this methodology, we don't require any memory elements for storing the symbols of Context Free Language (CFL) and Linear Bounded Automata (LBA), it's the main key factor of this method. One major point is that, In the anatomy of Context Free Language, generally Push Down Automata (PDA) are used to validate or to verify any CFL, using the memory element Stack (LIFO) which contains or to consume memory in a very large manner and similarly for Context Sensitive Language (CSL) a Linear Bounded Automata (LBA) are used to validate any CSL which used two Stacks (LIFO) as a memory elements. In this newly proposed method, no memory elements are required to validate any Context Free Language (CFL) and Context Sensitive Language (CSL). This method accepts both Context Free Language (CFL) and Context Sensitive Language (CSL). The mechanism of this method are as follows:
Conceptual Theory: This method are as same as the construction of Deterministic Finite Automata (DFA), but a one main difference is that this state machines contains some essential parameters or notations which makeup this MRA state machines are as follows:

$\{q_0, \sum, F, Q, \delta, \psi\}$
Here $q_0 = Starting\ State$
$\sum = Input\ Symbols$
$F = Final\ State$
$Q = Number\ of\ Finite\ States$
$\delta = Transition\ Function$
$\psi = Progressive\ Symbol$

These are some basic parameters of MRA.
Generally ψ (progressive symbol) are used to count the occurring patterns of any particular symbol.
Some illustration of this MRA are as follows:
The MRA of this CFL $a^n b^n c^n$ are as follows:

| e-ISSN: 2319-8753, p-ISSN: 2347-6710| www.ijirset.com | Impact Factor: 7.569|

|| Volume 10, Issue 8, August 2021 ||

| DOI:10.15680/IJIRSET.2021.1008104 |

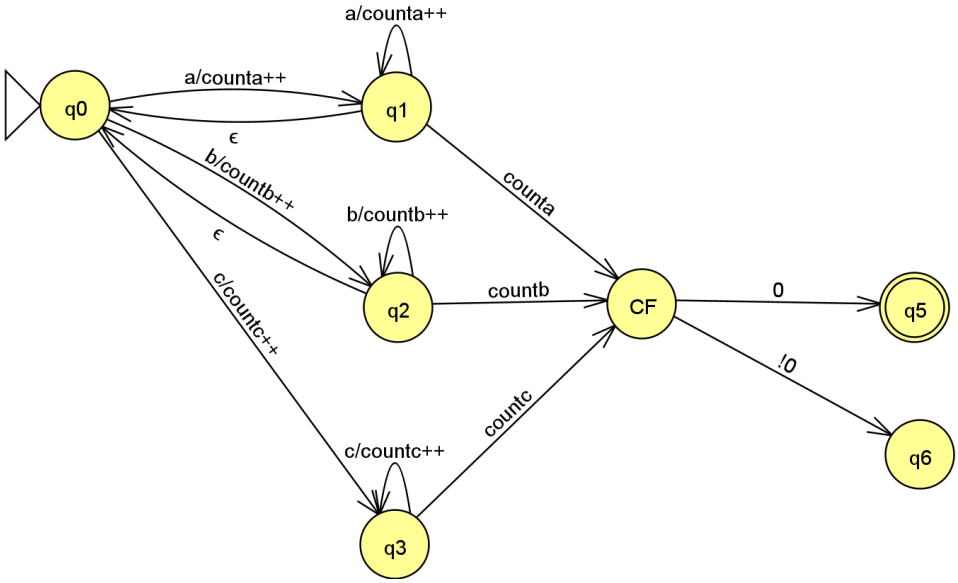

Fig d: MRA of $a^n b^n c^n$ for $n = 3$.

In this Mutative Regulative Automata (MRA) we validate following $a^n b^n c^n$ CSL.
In this diagrammatic representation after encounter first 'a' it's increment $counta$ variable value by 1, which acts as a progressive symbol ψ and responsible for counting the total occurrences of a's and reached to the next state which is q_1 state and again if any a's encountered its repeats this process again and again and increment $counta$ variable values by 1. After encountered first 'b' its increment $countb$ variable value by 1 and reached to the next state q_2 and for next remaining b's its repeats this process again and again and increment $countb$ variable value by 1. After encountered first 'c' it's increment $countc$ variable value by 1 and reached to the next state which is q_3 and for remaining c's it's repeat this process again and again and increments $countc$ variable values by 1, Now after occur the finalize values of all progressive variables ψ such as $\psi\{counta, countb, countc\}$ for respective symbols a, b, c a CF (Control Function) are responsible to do further calculations:

$$CF\ (Counting\ Function) = \sum_{i=0}^{m} (\psi_i) - (n * m)$$

Here m represents total number of distinct symbols are existed in a CFL,
and n represents total number of occurrences of particular symbol in CFl.
In this references $\psi = \{counta, countb, countc\}$
Here $counta = 3, countb = 3, countc = 3$
and $m = 3, so\ (counta + countb + countc) = (3 + 3 + 3) = 9$
and this MRA for this language $a^n b^n c^n$ *and here* $n = 3, so\ a^3 b^3 c^3$
Now $(n * m) = (3 * 3) = 9$
then $CF = (9 - 9) = 0$

Now if counting Function (CF) generate 0 value after counting, then it's reached to the final state q_5, and the CFL language will be accepted, but if any non-zero (!0) value occurred through counting Function (CF), then it's reached to the dead state q_6 and the CFL Language will be rejected.
So this is the main concept of MRA first method.

Second Method:
This is another a new proposed method or In other words a new alternative method for designing MRA. This methods also capable for accepting both Context Free Language (CFL) and Context Sensitive Language (CSL), but both these methods an essential memory element required for respectively PDA and LBA, generally PDA used one Stack (LIFO) for memory element and LBA used two Stacks (LIFO) for memory elements, but In this newly alternate proposed method again we don't require any memory elements.

| e-ISSN: 2319-8753, p-ISSN: 2347-6710| www.ijirset.com | Impact Factor: 7.569|

|| Volume 10, Issue 8, August 2021 ||

| DOI:10.15680/IJIRSET.2021.1008104 |

Some essential Parameters of this state machine are as follows:

$\{q_0, \sum, F, Q, \delta, \text{ॐ}\}$
Here $q_0 = Starting\ State$
$\sum = Input\ Symbols$
$F = Final\ State$
$Q = Number\ of\ Finite\ States$
$\delta = Transition\ Function$
$\text{ॐ} = Non\ discrimination\ Symbol$

These are some basic parameters of an alternative method of MRA.
This methods or alternative state machines accepted both Context Sensitive Language (CSL) and Context Free Language (CFL).
The diagrammatic representation of this CSL Language $a^n b^n c^n\ for\ n = 3$ are as follows:

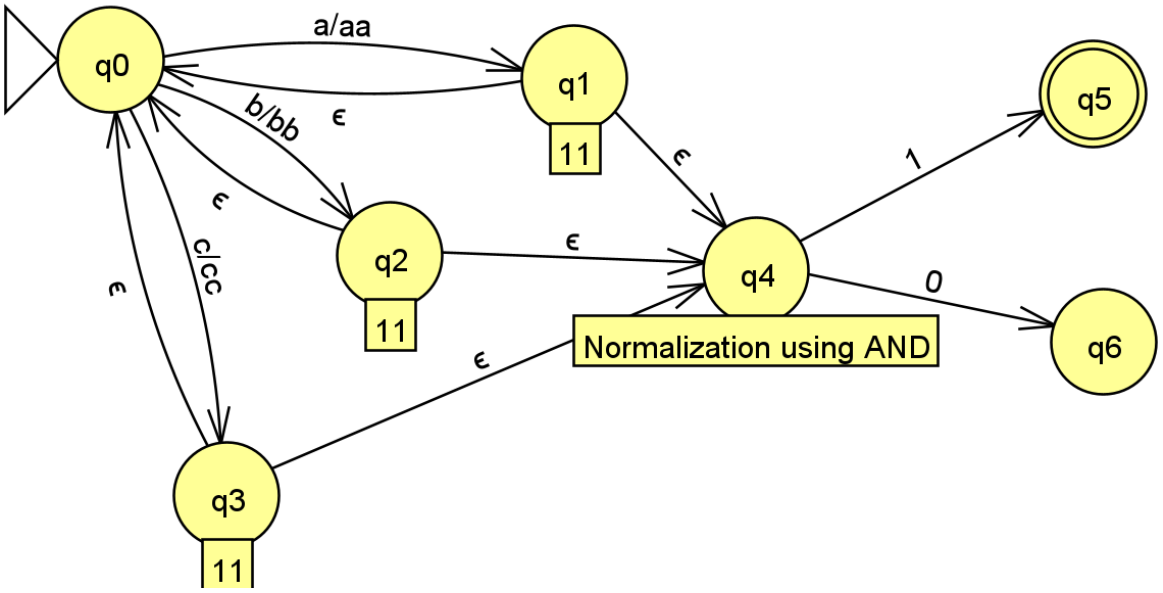

Fig e: An alternative MRA for Language $a^n b^n c^n\ for\ n = 3$.

In this methodology for language $a^n b^n c^n\ for\ n = 3$ after encountered first 'a' the non-discrimination symbol defines the remaining a's and it's reached to the q_1 state, In the q1 state equality operation performed between the current symbol and remaining symbols of that currently symbols , if the symbols are equal, then it's produced value '1 s', otherwise '0 s'. In this references

a/aa
$then\ (a = a) \rightarrow 1$
$and\ again\ (a = a) \rightarrow 1$
$so\ two\ one's\ (11)\ are\ produced\ so\ q_1\ contains\ 11.$

Similarly after encountered first 'b' it's reached to the next state q_2 and non-discrimination symbol defines the number of remaining b's. At the state q_2 equality operations performed between the current symbols and the remaining symbols of current symbols. If the Symbol are equal then it's produced value '1 s' otherwise value '0 s'. In this references:

b/bb
$then\ (b = b) \rightarrow 1$
$and\ again\ (b = b) \rightarrow 1$
$so\ two\ one's\ (11)\ are\ produced\ so\ q_2\ contains\ 11.$

| e-ISSN: 2319-8753, p-ISSN: 2347-6710| www.ijirset.com | Impact Factor: 7.569|

|| Volume 10, Issue 8, August 2021 ||

| DOI:10.15680/IJIRSET.2021.1008104 |

Similarly after encountered first 'c' , it's reached to the q_3 state and the non-discrimination symbol defines the remaining c's and at the state q_3 equality operations will performed between the current symbols and the remaining symbols of the current symbols, if the values are equal then 1's will produced, otherwise 0's. In this references:

c/cc
$then\ (c = c) \rightarrow 1$
$and\ again\ (c = c) \rightarrow 1$
$so\ two\ one's\ (11)\ are\ produced\ so\ q_3\ contains\ 11.$

After performing the equality operations for all symbols, then it will reached to the q_4 state, where the normalized functions perform the normalization operation. In this normalized operation LOGICAL AND will performed between the equality value which is {11,11,11} for respective states $\{q_1, q_2, q_3\}$, Such as:

$q1 \rightarrow 11, q2 \rightarrow 11, q3 \rightarrow 11$

$q1 \rightarrow 11$
$q2 \rightarrow 11$
$q3 \rightarrow 11$
$then\ (1\ AND\ 1\ AND\ 1) \rightarrow 1\ for\ first\ series\ of\ 1's.$
$then\ (1\ AND\ 1\ AND\ 1) \rightarrow 1\ for\ second\ series\ of\ 1's.$
$Now\ we\ get\ two\ resultant\ 1's\ and\ again\ perform\ AND$
$operation\ between\ these\ two\ resultant\ 1's.$
$1\ AND\ 1 \rightarrow 1.$

Now if '1' are produced from state q_4 then it's reached to the final state q_5 and the language are accepted otherwise if '0' are produced then it's reached to the dead state and the language will be rejected.

So this is the mechanism of this alternative method of MRA state machine.

IV. FUTURISTIC SCOPE AND APPLICATIONS

Mutative Regulative Automata (MRA) are one of the major state machine which are responsible to accept Context Sensitive Language (CSL) and Context Free Language (CFL). Generally Linear Bounded Automata (LBA) and Push Down Automata (PDA) are responsible to accept Context Sensitive Language (CSL) and Context Free Language (CFL) respectively, now a one major point is that Linear Bounded Automata (LBA) and Push Down Automata (PDA) are used a memory elements to validate any Context Sensitive Language (CSL) and Context Free Language (CFL). Generally Linear Bounded Automata (LBA) used two Stacks (LIFO) type of data structure to validate any Context Sensitive Language and Push Down Automata (PDA) used a one Stack (LIFO) type of data structure to validate any Context Free Language (CFL), but In this newly proposed Mutative Regulative Automata (MRA) does not contains any memory element to validate any Context Sensitive Language (CSL) and Context Free Language (CFL). Mutative Regulative Automata (MRA) are the fusion of Linear Bounded Automata (LBA) and Push Down Automata (PDA), It's a hybrid approached based state machine, which are responsible to accept both Context Sensitive Language (CSL) and Context Free Language (CFL). That's why it's a very strong state machine, which are responsible CSL and CFL, where previously two different state machines LBA and PDA are required to validate it. Generally here we proposed two novel method to designed Mutative Regulative Automata (MRA), both the methods are same or to do perform similar type of tasks, but the way of designing are different, in the first method progressive symbol play's a major role. This progressive symbol are responsible to do the main task of first method, and in the secondly method non-discrimination symbol are responsible to do the main task of second method.

This Mutative Regulative Automata (MRA) also reliable worked with some Real Time applications and Computational Theory. MRA does not contains any memory element, so the essential time required to do or perform the computations on any state are also optimized. So it's time and space complexity are much lesser as compare to other previously developed traditional approach based state machines. Now its works very reliable and appropriately in manner and gives a rapid response on any state and its throughput also becomes very high and computation Cost are minimum.

Some of the major identical areas of this Mutative Regulative Automata (MRA) are as follows:

- ✓ Automata Theory
- ✓ Computability Theory

| e-ISSN: 2319-8753, p-ISSN: 2347-6710| www.ijirset.com | Impact Factor: 7.569|

|| Volume 10, Issue 8, August 2021 ||

| DOI:10.15680/IJIRSET.2021.1008104 |

- ✓ Computational Complexity Theory

These are some areas where Mutative Regulative Automata (MRA) plays a vital or significant role.
Computational Complexity Theory contains some various real life problems where this Mutative Regulative Automata (MRA) play's a significant role, some of these areas/problems are as follows:

- ✓ Lambda Calculus [8]
- ✓ Combinatory Logic [9]
- ✓ Markov Algorithm [8]
- ✓ Register Machines [8,11]

So these are some areas where Mutative Regulative Automata (MRA) are one of the most fundamentally strong and Technical Computational Rich state machine.
MRA computation cost are very lesser compare to other state machines such as LBA and PDA which accepts Context Sensitive Language (CSL) and Context Free Language (CFL).

V. CONCLUSION

Mutative Regulative Automata (MRA) are a versatile State Machines, which accepts both the Context Sensitive Language (CSL) and Context Free Language (CFL), generally it's a fusion of both CSL and CFL. One main highlighted key-features of this state machine is that it's does not contains any memory elements. In the context of Linear Bounded Automata (LBA), which is responsible for accepted Context Sensitive Language (CSL) required two stacks (LIFO) data Structure as a memory elements to validate any Context Sensitive Language (CSL), and similarly In the Push Down Automata (PDA) which is generally responsible for validate any Context Free Language (CFL) required a one Stack (LIFO) data structure as a memory element, but In the references of this novel Mutative Regulative Automata (MRA) no such types of memory elements required to validate any CSL and CFL. That's why it's execution time and the reliability of this state machines are much more superior than other previously state machines, which based on some traditional approach. Delay Between the states during execution to conduct any concurrent executive processes relatively very low as compared to Linear Bounded Automata (LBA) and Push Down Automata (PDA). Generally two different state machines are required to validate any Context Sensitive Language (CSL) and Context Free Language (CFL), but here in this context Mutative Regulative Automata (MRA) are a hybrid state machines, which accepted both Context Sensitive Language (CSL) and Context Free Language (CFL) using only one state machine, no separate state machine are required to validate any Context Sensitive Language (CSL) and Context Free Language (CFL). This Mutative Regulative Automata (MRA) response time much faster than Linear Bounded Automata (LBA) and Push Down Automata (PDA). Here we proposed two new methods to designed a novel Mutative Regulative Automata (MRA), which is much reliable and efficient to compute any typical information. This MRA are more valuable for Real Time systems , because In the Context of LBA and PDA memory as an essential elements for them, and that's why it's throughput and response time are much slower relatively Mutative Regulative Automata. To designed any Mutative Regulative Automata (MRA) here we proposed two methods, In the first method progressive symbol are the main symbol, which is responsible to do the main task of first method, and secondly In the alternative method non-discrimination symbol are the main symbol, which is also responsible to do the main task of second method. There are various key features of MRA and responsible to accepted or validate any Context Sensitive Language (CSL) and Context Free Language (CFL).

VI. ACKNOWLEDGEMENTS

We are very much grateful to all respected professors of DIET Rishikesh for their kind help, lasting encouragement, valuable suggestion throughout the entire period of our project work. We are highly indebted to his astute guidance, sincere support and boosting confidence to make this Research successful.
The acknowledgment will be incomplete if we fail to express our obligation and reverence to our family members and friends whose moral support is great factor in doing this research.

VII. FUNDING

This study was not funded by any other profitable or non-profitable organisations.

| e-ISSN: 2319-8753, p-ISSN: 2347-6710| www.ijirset.com | Impact Factor: 7.569|

|| Volume 10, Issue 8, August 2021 ||

| DOI:10.15680/IJIRSET.2021.1008104 |

VIII. CONFLICT OF INTEREST

The authors declare that they have no conflict of interest.

REFERENCES

[1] Yan, Song Y. “An Introduction to Formal Languages and Machine Computation. Singapore”. World Scientific Publishing Co. Pte. Ltd. pp. 155–156, 1998.
[2] Chakraborty, P., Saxena, P. C., Katti, C. P. 2011. Fifty Years of Automata Simulation: A Review. ACM Inroads, 2(4):59–70. http://dl.acm.org/citation.cfm?id=2038893&dl=ACM&coll=DL&CFID=65021406&CFTOKEN=86634854
[3] Jirí Adámek and Vera Trnková. 1990. “Automata and Aslgebras in Categories”. Kluwer Academic Publishers:Dordrecht and Prague
[4] S. Mac Lane, Categories for the Working Mathematician, Springer, New York 1971 .
[5] Cartesian closed category Archived November 16, 2011, at the Wayback Machine.
[6] The Category of Automata Archived September 15, 2011, at the Wayback Machine.
[7] http://www.math.cornell.edu/~worthing/asl2010.pdf James Worthington.2010.Determinizing, Forgetting, and Automata in Monoidal Categories. ASL North American Annual Meeting, March 17, 2010
[8] Aguiar, M. and Mahajan, S.2010. "Monoidal Functors, Species, and Hopf Algebras".
[9] Meseguer, J., Montanari, U.: 1990 Petri nets are monoids. Information and Computation 88:105–155
[10] S. B. Cooper, 2004. Computability Theory, Chapman & Hall/CRC. ISBN 1-58488-237-9
[11] N. Cutland, 1980. Computability, An introduction to recursive function theory, Cambridge University Press. ISBN 0-521-29465-7
[12] Y. Matiyasevich, 1993. Hilbert's Tenth Problem, MIT Press. ISBN 0-262-13295-8

INNO SPACE
SJIF Scientific Journal Impact Factor

Impact Factor:
7.569

doi cross ref

ISSN INTERNATIONAL STANDARD SERIAL NUMBER INDIA

निस्केयर NISCAIR

INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH

IN SCIENCE | ENGINEERING | TECHNOLOGY

9940 572 462 6381 907 438 ijirset@gmail.com

www.ijirset.com

Scan to save the contact details