I.J. Mathematical Sciences and Computing, **2019, 1, 65-79**
Published Online January 2019 in MECS (http://www.mecs-press.net)
DOI: 10.5815/ijmsc.2019.01.06

Available online at **http://www.mecs-press.net/ijmsc**

A New Approach to the Design of a Finite Automaton that accepts Class of IPV_4 Addresses

P.Sri Ram Chandra [1*], K.S.Sravan [2], M.S.Chakravarthy [3]

[1, 2, 3] *Department of Computer Science and Engineering, Godavari Institute of Engineering and Technology (A), Rajahmundry, Andhra Pradesh, India.*

Received: 19 April 2018; Accepted: 06 August 2018; Published: 08 January 2019

Abstract

Theory of computation is characterized as calculation through the conceptual machines. The three essential unique machines utilized are Finite Automata, Pushdown Automata and Turing Machine. In this paper we propose an outline of Finite Automata that accepts the class of IPV_4 Addresses.

Index Terms: Finite Automata, IP Address.

1. Introduction

Theory of computation is defined as computation through the abstract machines. Finite Automata is one among the primary abstract machines. It is represented as a five tuple machine 'M'.

$M = (Q, \sum, \delta, S, F)$
Q – Finite Set of States
$\sum$ – Finite Set of Input Symbols (alphabets)
δ – Transition Function ($Q \times \sum \rightarrow Q$)
S – Initial State ($S \in Q$)
F – Final State ($F \subseteq Q$)

* Corresponding author.
E-mail address: psrgietcse@gmail.com

1.1. Alphabets

An alphabet is a finite, nonempty set of inputs usually denoted as $\sum$.
Examples of the alphabets

- Binary alphabet $\sum = \{0, 1\}$
- English alphabet$\sum = \{a, b, \ldots, z\}$

Strings

A string (or word) is a finite sequence of symbols from an alphabet. For example --- 1011 is a string from the binary alphabet $\sum = \{0, 1\}$.

- Empty string ‘ ’is defined as a string with zero occurrences of symbols.
- Length ‘|W|’ of string ‘W’ is defined as the number of positions for symbols in W. For example length of the string ‘0111’ is $|0111| = 4$, Length of the string ‘ ’ is $|\ | = 0$.
- Set of all strings over $\sum$ is usually denoted as $\sum^*$ i.e., $\sum^* = \sum^0 \cup \sum^1 \cup {}^{*2} \cup \ldots \sum^n$
- $\sum^+$ = the set of nonempty strings from $\sum = \sum^* - \{\ \}$. Therefore, we have

$$\sum^+ = \sum^1 \cup \sum^2 \cup \sum^3 \cup \ldots \sum^n \text{ and } \sum^* = \sum^+ \cup \{\varepsilon\}$$

Languages

A language is a set of strings all chosen from some $\sum^*$. In other words, if $\sum$ is an alphabet, and $L \subseteq \sum^*$, then L is a language over $\sum$.
Examples:

- For the language L: Set of all legal English words, $\sum$ = {the set of all letters}
- For the language L: Program written in ‘C’ Language, $\sum$ = { a subset of the ASCII characters}
- The set of all strings of n 0’s followed by n 1’s for $n \geq 0$: $\{\varepsilon, 01, 0011, 000111, \ldots\}$
- $\sum^*$ is an infinite language for any alphabet $\sum$.
- $\emptyset$ Denotes the empty language (not the empty string ε) which is a language over any alphabet.
- $\{\varepsilon\}$ is a language over any alphabet (consisting of only one string, the empty string ε).

Different ways of describing languages

Description by exhaustive listing

- $L_1 = \{a, ab, abc\}$ (finite language; listed one by one)
- $L_2 = \{a, ab, abb, abbb, ...\}$ (infinite language; listed partially)
- $L_3 = L(ab^*)$ (infinite language; expressed by a regular expression)

Description by generic elements

- $L_4 = \{x \mid x \text{ is over } V = \{a, b\}, \text{ begins with a, followed by any number of b, possible none}\}$
- Note: $L_4 = L_3 = L_2$

Description by integer parameters

- $L_5 = \{ab^n \mid n \geq 0\}$
- Note: $L_5 = L_4 = L_3 = L_2$

Closures of language L:

- Positive closure of a language L is usually denoted as $L^+ = L^1 \cup L^2 \cup \ldots L^n$
- Star Closure of a language L is usually denoted as $L^* = \bigcup_{i=0}^{\infty} L^i = L^0 \cup L^1 \cup L^2 \cup \ldots L^n$
- Note: $L^* - L^0 = L^* - \{\varepsilon\}$

1.2. Internet Protocol (IP) Addresses and Automata

The key contribution to this research paper starts with the basic idea about IPV_4 addresses and identifying the required relationships between IPV_4 address and Constructing Finite Automata. Usually the communication between the computers can be done through the network. For any level of communication we need a global addressing scheme i.e., IP addressing scheme to mean a logical address in the network layer of the TCP/IP protocol suite. These addresses are referred to as IPV_4 addresses. An IPV_4 address is a 32 bit address that uniquely defines the connection of a device through Internet. The address space of the IPV_4 address is 2^{32} addresses (4,294,967,296 addresses are available). An IPV_4 address can be represented as binary notation and dotted-decimal notation. The following Fig. 1 depicts the notations of the IPV_4 address.

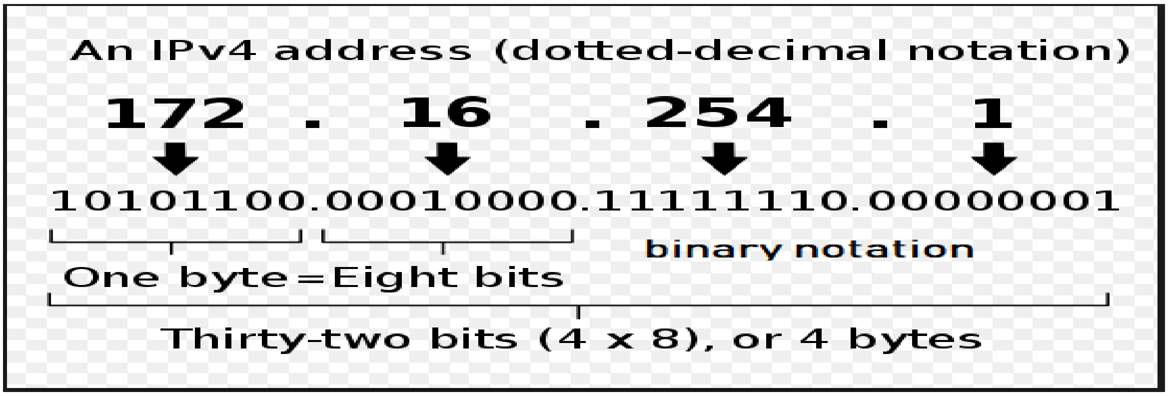

Fig.1. Binary and Dotted-Decimal Notations of the IPV_4 address.

1.3. Classful Addressing

Since its inception, the IPV_4 addressing used the concept of classes. In classful addressing, the address space is divided into five classes namely A, B, C, D and E. Each class occupies some part of the address space. We can find the class of an address when it is represented in binary or dotted-decimal notations. The first few bits in binary notation of IP address will tell us the class name. The first byte in dotted-decimal notation of IP address also tells us the class. The following Fig. 2 depicts both the methods. As far as the finite automaton is concerned we use only binary notation of IP address to detect its class.

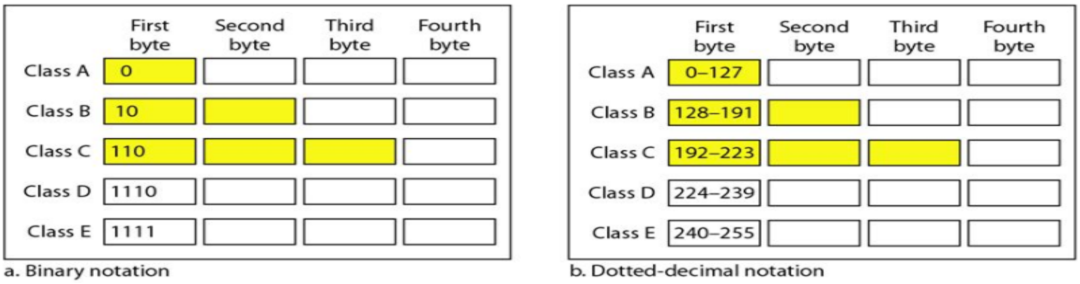

Fig.2. Methods to detect the class of IP Address

The Fig. 2 clearly depicts that, first few bits in binary notation of IPV_4 address plays a vital role in detecting its respective class. As the first bit (MSB) of Class A in Binary notation is '0', we can say that it is the string of 32 bits [actual length of IPV_4 address]. According to the Theory of Automata, the language for the Class A address is designated as $L = \{0\ W|W \in (0 + 1)^*\}$. Similarly the language for rest of the classes is as follows:

Language to detect Class B IP address: $L = \{10\ W|W \in (0 + 1)^*\}$
Language to detect Class C IP address: $L = \{110\ W|W \in (0 + 1)^*\}$
Language to detect Class D IP address: $L = \{1110\ W|W \in (0 + 1)^*\}$
Language to detect Class E IP address: $L = \{1111\ W|W \in (0 + 1)^*\}$

2. Construction of Finite Automata

In this section we describe the construction of Finite Automata based on the corresponding languages defined earlier.

2.1. Design of a Finite Automata to accept Class 'A' IP address

According to the Fig. 2, the first bit of Class A in Binary notation is '0', which indicates that it is a string of 32 bits [actual length of IPV4 address]. Thus the language for the Class 'A' address is designated as

$$L = \{0\ W|W \in (0 + 1)^*\}$$

or

$$L = \{0 \text{ followed by any number of 0s and any number of 1s}\}.$$

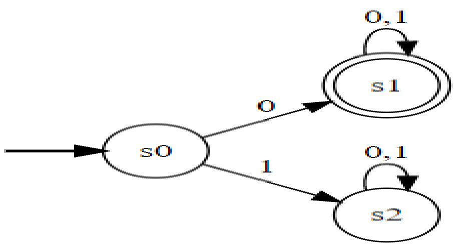

Fig.3. Transition Diagram of a Finite Automata the accepts the Class 'A' IP Addresses

The input alphabets are denoted as $\sum = \{0,1\}$ and the corresponding Regular Expression is given as: $0(0+1)^*$. The following Fig. 3 depicts the corresponding automata.

Finite Automata are often represented by digraphs called **transition diagram**. The vertices (denoted by single circles) of a transition diagram represents various states of the Finite Automata and the edges labelled with an input symbol correspond to the transitions. An edge (0, 1) from state s_0 to state s_1 with label '0' represents the transition $\delta(s_0, 0) = s_1$. The acceptance states or final states are depicted by double circles. Transition functions can also be represented by tables as shown in the table1called as transition table.

Table 1. Transition Table for the Finite Automata shown in Fig. 3

	$\sum$	
Q	**0**	**1**
s_0	s_1	s_2
s_1	s_1	s_1
s_2	s_2	s_2

2.2. Design of a Finite Automata to accept Class 'B' IP address

In the Fig. 2, the first two bits of Class 'B' in Binary notation are '10', means that it is the string of 32 bits [actual length of IPV4 address] where the two bits from MSB (including) are confined to '10'. Thus the language for the Class B address is designated as

$$L = \{10\ W | W \in (0+1)^*\}$$

or

$$L = \{10 \text{ followed by any number of 0s and any number of 1s}\}.$$

The input alphabets are denoted as $\sum = \{0,1\}$ and the corresponding Regular Expression is given as: $10(0+1)^*$. The following Fig. 4 depicts the corresponding automata.

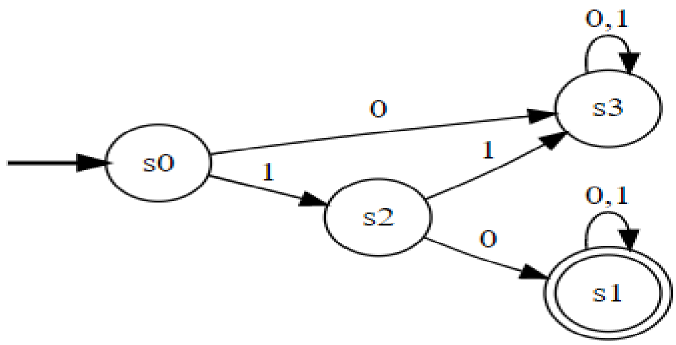

Fig.4 Transition Diagram of a Finite Automata the accepts the Class 'B' IP Addresses

Table 2. Transition Table for the Finite Automata shown in Fig. 4

Q	$\sum$	
	0	**1**
s_0	s_3	s_2
s_1	s_1	s_1
s_2	s_1	s_3
s_3	s_3	s_3

2.3. Design of a Finite Automata to accept Class 'C' IP address

As shown in the Fig. 2, the first two bits of Class C in Binary notation are '110', means that it is a string of 32 bits [actual length of IPV4 address] where the three bits from MSB are confined to '110'. Thus the language for the Class C address is designated as

$$L = \{110\ W | W \in (0+1)^*\}$$

or

$$L = \{110 \text{ followed by any number of 0s and any number of 1s}\}.$$

The input alphabets are denoted as $\sum = \{0,1\}$ and the corresponding Regular Expression is given as: $110(0+1)^*$. The following Fig. 5 depicts the corresponding automata.

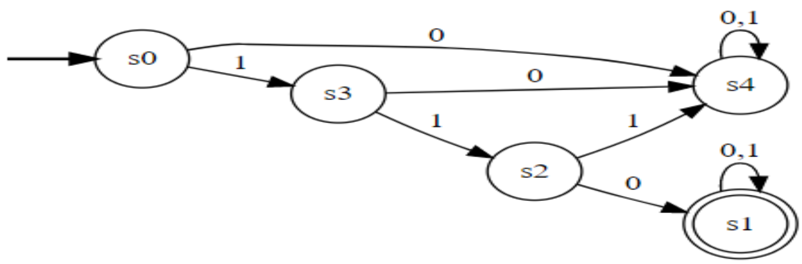

Fig.5. Transition Diagram of a Finite Automata the accepts the Class 'C' IP Addresses

Table 3. Transition Table for the Finite Automata shown in Fig. 5

	$\sum$	
Q	**0**	**1**
s_0	s_4	s_3
s_1	s_1	s_1
s_2	s_1	s_4
s_3	s_4	s_2
s_4	s_4	s_4

2.4. Design of a Finite Automata to accept Class 'D' IP address

As shown in the Fig. 2, the first two bits of Class D in Binary notation are '1110', means that it is a string of 32 bits [actual length of IPV4 address] where the four bits from MSB are confined to '1110'. Thus the language for the Class D address is designated as

$$L = \{1110\ W | W \in (0+1)^*\}$$

or

$$L = \{1110 \text{ followed by any number of 0s and any number of 1s}\}.$$

The input alphabets are denoted as $\sum = \{0,1\}$ and the corresponding Regular Expression is given as: $1110(0+1)^*$. The following Fig. 5 depicts the corresponding automata.

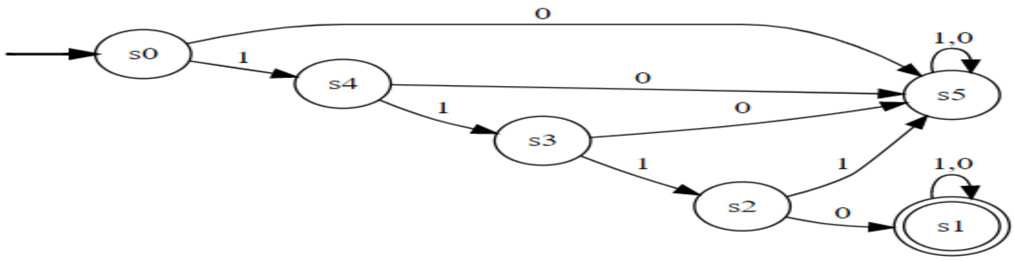

Fig.6. Transition Diagram of a Finite Automata the accepts the Class 'D' IP Addresses

Table 4. Transition Table for the Finite Automata shown in Fig. 6

	$\sum$	
Q	**0**	**1**
s_0	s_5	s_4
s_1	s_1	s_1
s_2	s_1	s_5
s_3	s_5	s_2
s_4	s_5	s_3
s_5	s_5	s_1

2.5. Design of a Finite Automata to accept Class 'E' IP address

As shown in the Fig. 2, the first two bits of Class E in Binary notation are '1111', means that it is a string of 32 bits [actual length of IPV4 address] where the four bits from MSB are confined to '1111'. Thus the language for the Class E address is designated as

$$L = \{1111\ W | W \in (0+1)^*\}$$

or

$$L = \{1111 \text{ followed by any number of 0s and any number of 1s}\}.$$

The input alphabets are denoted as $\sum = \{0,1\}$ and the corresponding Regular Expression is given as: $1110(0+1)^*$. The following Fig. 5 depicts the corresponding automata.

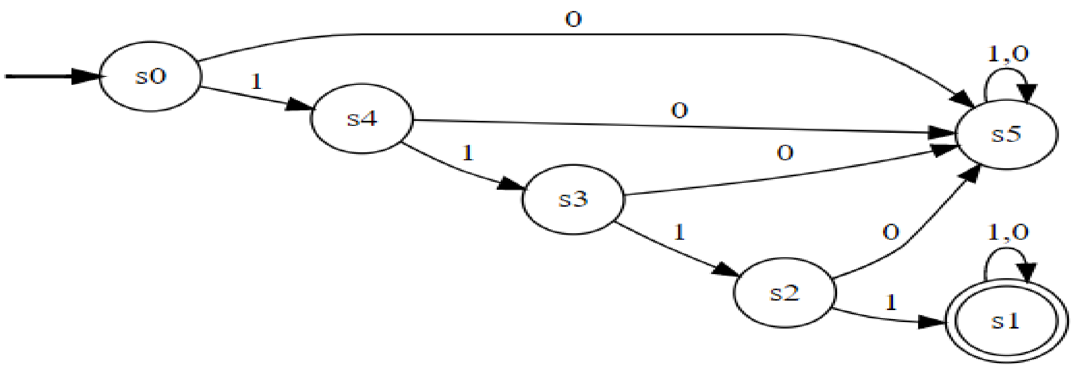

Fig.7. Transition Diagram of a Finite Automata the accepts the Class 'E' IP Addresses

Table 5. Transition Table for the Finite Automata shown in Fig. 7

	$\sum$	
Q	**0**	**1**
s_0	s_5	s_4
s_1	s_1	s_1
s_2	s_5	s_1
s_3	s_5	s_2
s_4	s_5	s_3
s_5	s_5	s_1

3. Results and Discussion

As far as the results section is concerned, we need to prove that the designed automata should accept the respective class of IP address as per the language mentioned. We can say that the automaton accepts the respective class of IP address if and only if "the validation of the considered 32 bit string starts at initial state and ends at final state". At first we will see the validation for Class 'A' IPV$_4$ addresses that is 32 bit string with MSB as '0', further we will see the rest of classes.

The input string to validate class 'A' IP Address:

IPV$_4$ Address	Table 6.a	Dotted Decimal Notation	79.111.127.99 [Should be Accepted]
		Binary Notation	01001111011011110111111101100011
	Table 6.b	Dotted Decimal Notation	207.111.69.102 [Should be Rejected]
		Binary Notation	11001111011011110100010101100110

In the table 6.a, the last state achieved through the automata is a final state. The automata is verified and hence proved that it accepts all the strings which start with ‘0’i.e., Class ‘A’ IPV4 Address. In the table 6.b, since the last state obtained is s2 and as it is not the final state in the automata, so the strings which start with other than ‘0’ are not accepted i.e., it accepts only Class ‘A’ IPV4 Address.

Table 6. Validation Table that shows the automata accepts Class a IPV 4 Address

Table 6.a						**Table 6.b**					
Current State-Q	Input-∑	Next State-Q	Current State-Q	Input-∑	Next State-Q	Current State-Q	Input-∑	Next State-Q	Current State-Q	Input-∑	Next State-Q
$Q \times \Sigma \rightarrow Q$			$Q \times \Sigma \rightarrow Q$			$Q \times \Sigma \rightarrow Q$			$Q \times \Sigma \rightarrow Q$		
s_0	0	s_1	s_1	0	s_1	s_0	1	s_2	s_2	0	s_2
s_1	1	s_1	s_1	1	s_1	s_2	1	s_2	s_2	1	s_2
s_1	0	s_1	s_1	1	s_1	s_2	0	s_2	s_2	0	s_2
s_1	0	s_1	s_1	1	s_1	s_2	0	s_2	s_2	0	s_2
s_1	1	s_1	s_1	1	s_1	s_2	1	s_2	s_2	0	s_2
s_1	1	s_1	s_1	1	s_1	s_2	1	s_2	s_2	1	s_2
s_1	1	s_1	s_1	1	s_1	s_2	1	s_2	s_2	0	s_2
s_1	1	s_1	s_1	1	s_1	s_2	1	s_2	s_2	1	s_2
s_1	0	s_1	s_1	0	s_1	s_2	0	s_2	s_2	0	s_2
s_1	1	s_1	s_1	1	s_1	s_2	1	s_2	s_2	1	s_2
s_1	1	s_1	s_1	1	s_1	s_2	1	s_2	s_2	1	s_2
s_1	0	s_1	s_1	0	s_1	s_2	0	s_2	s_2	0	s_2
s_1	1	s_1	s_1	0	s_1	s_2	1	s_2	s_2	0	s_2
s_1	1	s_1	s_1	0	s_1	s_2	1	s_2	s_2	1	s_2
s_1	1	s_1	s_1	1	s_1	s_2	1	s_2	s_2	1	s_2
s_1	1	s_1	s_1	1	s_1	s_2	1	s_2	s_2	0	s_2

The input string to validate class ‘B’ IP Address:

IPV_4 Address	Table 7.a	Dotted Decimal Notation	143.111.127.99 [Should be Accepted]
		Binary Notation	10001111011011110111111101100011
	Table 7.b	Dotted Decimal Notation	207.111.69.102 [Should be Rejected]
		Binary Notation	11001111011011110100010101100110

In the table 7.a, the last state achieved through the automata is a final state. The automata is verified and hence proved that it accepts all the strings which start with ‘10’i.e., Class ‘B’ IPV_4 Address. In the table 7.b, since the last state obtained is s_3 and as it is not the final state in the automata, so the strings which start with other than ‘10’ are not accepted i.e., it accepts only Class ‘B’ IPV_4 Address.

Table 7. Validation table that shows the Automata accepts Class ‘B’ IPV4 Address

Table 7.a						**Table 7.b**					
Current State-Q	Input-∑	Next State-Q	Current State-Q	Input-∑	Next State-Q	Current State-Q	Input-∑	Next State-Q	Current State-Q	Input-∑	Next State-Q
Q X ∑→Q			Q X ∑→Q			Q X ∑→Q			Q X ∑→Q		
s_0	1	s_2	s_1	0	s_1	s_0	1	s_2	s_3	0	s_3
s_2	0	s_1	s_1	1	s_1	s_2	1	s_3	s_3	1	s_3
s_1	0	s_1	s_1	1	s_1	s_3	0	s_3	s_3	0	s_3
s_1	0	s_1	s_1	1	s_1	s_3	0	s_3	s_3	0	s_3
s_1	1	s_1	s_1	1	s_1	s_3	1	s_3	s_3	0	s_3
s_1	1	s_1	s_1	1	s_1	s_3	1	s_3	s_3	1	s_3
s_1	1	s_1	s_1	1	s_1	s_3	1	s_3	s_3	0	s_3
s_1	1	s_1	s_1	1	s_1	s_3	1	s_3	s_3	1	s_3
s_1	0	s_1	s_1	0	s_1	s_3	0	s_3	s_3	0	s_3
s_1	1	s_1	s_1	1	s_1	s_3	1	s_3	s_3	1	s_3
s_1	1	s_1	s_1	1	s_1	s_3	1	s_3	s_3	1	s_3
s_1	0	s_1	s_1	0	s_1	s_3	0	s_3	s_3	0	s_3
s_1	1	s_1	s_1	0	s_1	s_3	1	s_3	s_3	0	s_3
s_1	1	s_1	s_1	0	s_1	s_3	1	s_3	s_3	1	s_3
s_1	1	s_1	s_1	1	s_1	s_3	1	s_3	s_3	1	s_3
s_1	1	s_1	s_1	1	s_1	s_3	1	s_3	s_3	0	s_3

The input string to validate class 'C' IP Address:

IPV_4 Address	Table 8.a	Dotted Decimal Notation	207.111.127.99 [Should be Accepted]
		Binary Notation	11001111011011110111111101100011
	Table 8.b	Dotted Decimal Notation	79.111.69.102 [Should be Rejected]
		Binary Notation	01001111011011110100010101100110

In the table 8.a, the last state achieved through the automata is a final state. The automata is verified and hence proved that it accepts all the strings which start with '110'i.e., Class 'C' IPV_4 Address. In the table 8.b, since the last state obtained is s_4 and as it is not the final state in the automata, so the strings which start with other than '110' are not accepted i.e., it accepts only Class 'C' IPV_4 Address.

Table 8. Validation table that shows the Automata accepts Class 'C' IPV4 Address

Table 8.a						**Table 8.b**					
Current State-Q	Input-∑	Next State-Q	Current State-Q	Input-∑	Next State-Q	Current State-Q	Input-∑	Next State-Q	Current State-Q	Input-∑	Next State-Q
Q X ∑→Q			Q X ∑→Q			Q X ∑→Q			Q X ∑→Q		
s_0	1	s_3	s_1	0	s_1	s_0	0	s_4	s_4	0	s_4
s_3	1	s_2	s_1	1	s_1	s_4	1	s_4	s_4	1	s_4
s_2	0	s_1	s_1	1	s_1	s_4	0	s_4	s_4	0	s_4
s_1	0	s_1	s_1	1	s_1	s_4	0	s_4	s_4	0	s_4
s_1	1	s_1	s_1	1	s_1	s_4	1	s_4	s_4	0	s_4
s_1	1	s_1	s_1	1	s_1	s_4	1	s_4	s_4	1	s_4
s_1	1	s_1	s_1	1	s_1	s_4	1	s_4	s_4	0	s_4
s_1	1	s_1	s_1	1	s_1	s_4	1	s_4	s_4	1	s_4
s_1	0	s_1	s_1	0	s_1	s_4	0	s_4	s_4	0	s_4
s_1	1	s_1	s_1	1	s_1	s_4	1	s_4	s_4	1	s_4
s_1	1	s_1	s_1	1	s_1	s_4	1	s_4	s_4	1	s_4
s_1	0	s_1	s_1	0	s_1	s_4	0	s_4	s_4	0	s_4
s_1	1	s_1	s_1	0	s_1	s_4	1	s_4	s_4	0	s_4
s_1	1	s_1	s_1	0	s_1	s_4	1	s_4	s_4	1	s_4
s_1	1	s_1	s_1	1	s_1	s_4	1	s_4	s_4	1	s_4
s_1	1	s_1	s_1	1	s_1	s_4	1	s_4	s_4	0	s_4

The input string to validate class ‘D’ IP Address:

IPV_4 Address	Table 9.a	Dotted Decimal Notation	239.111.127.99 [Should be Accepted]
		Binary Notation	11101111011011110111111101100011
	Table 9.b	Dotted Decimal Notation	202.69.42.171 [Should be Rejected]
		Binary Notation	11001010010001010010101010101011

In the table 9.a, the last state achieved through the automata is a final state. The automata is verified and hence proved that it accepts all the strings which start with ‘1110’i.e., Class ‘D’ IPV_4 Address. In the table 9.b, since the last state obtained is s_5 and as it is not the final state in the automata, so the strings which start with other than ‘1110’ are not accepted i.e., it accepts only Class ‘D’ IPV_4 Address.

Table 9. Validation table that shows the Automata accepts Class ‘D’ IPV4 Address

Table 9.a						**Table 9.b**					
Current State-Q	Input-∑	Next State-Q	Current State-Q	Input-∑	Next State-Q	Current State-Q	Input-∑	Next State-Q	Current State-Q	Input-∑	Next State-Q
Q X ∑→Q			Q X ∑→Q			Q X ∑→Q			Q X ∑→Q		
s_0	1	s_4	s_1	0	s_1	s_0	1	s_4	s_5	0	s_5
s_4	1	s_3	s_1	1	s_1	s_4	1	s_3	s_5	0	s_5
s_3	1	s_2	s_1	1	s_1	s_3	0	s_5	s_5	1	s_5
s_2	0	s_1	s_1	1	s_1	s_5	0	s_5	s_5	0	s_5
s_1	1	s_1	s_1	1	s_1	s_5	1	s_5	s_5	1	s_5
s_1	1	s_1	s_1	1	s_1	s_5	0	s_5	s_5	0	s_5
s_1	1	s_1	s_1	1	s_1	s_5	1	s_5	s_5	1	s_5
s_1	1	s_1	s_1	1	s_1	s_5	0	s_5	s_5	0	s_5
s_1	0	s_1	s_1	0	s_1	s_5	0	s_5	s_5	1	s_5
s_1	1	s_1	s_1	1	s_1	s_5	1	s_5	s_5	0	s_5
s_1	1	s_1	s_1	1	s_1	s_5	0	s_5	s_5	1	s_5
s_1	0	s_1	s_1	0	s_1	s_5	0	s_5	s_5	0	s_5
s_1	1	s_1	s_1	0	s_1	s_5	0	s_5	s_5	1	s_5
s_1	1	s_1	s_1	0	s_1	s_5	1	s_5	s_5	0	s_5
s_1	1	s_1	s_1	1	s_1	s_5	0	s_5	s_5	1	s_5
s_1	1	s_1	s_1	1	s_1	s_5	1	s_5	s_5	1	s_5

The input string to validate class ‘E’ IP Address:

IPV_4 Address	Table 10.a	Dotted Decimal Notation	245.85.82.170 [Should be Accepted]
		Binary Notation	11110101010101010101001010101010
	Table 10.b	Dotted Decimal Notation	174.158.162.174 [Should be Rejected]
		Binary Notation	10101110100111101010001010101110

In the table 10.a, the last state achieved through the automata is a final state. The automata is verified and hence proved that it accepts all the strings which start with ‘1111’i.e., Class ‘E’ IPV_4 Address. In the table 10.b, since the last state obtained is s_3 and as it is not the final state in the automata, so the strings which start with other than ‘1111’ are not accepted i.e., it accepts only Class ‘E’ IPV_4 Address.

Table 10. Validation table that shows the Automata accepts Class ‘E’ IPV4 Address

Table 10.a						**Table 10.b**					
Current State-Q	Input-∑	Next State-Q	Current State-Q	Input-∑	Next State-Q	Current State-Q	Input-∑	Next State-Q	Current State-Q	Input-∑	Next State-Q
Q X ∑→Q			Q X ∑→Q			Q X ∑→Q			Q X ∑→Q		
s_0	1	s_4	s_1	0	s_1	s_0	1	s_4	s_5	1	s_5
s_4	1	s_3	s_1	1	s_1	s_4	0	s_5	s_5	0	s_5
s_3	1	s_2	s_1	0	s_1	s_5	1	s_5	s_5	1	s_5
s_2	1	s_1	s_1	1	s_1	s_5	0	s_5	s_5	0	s_5
s_1	0	s_1	s_1	0	s_1	s_5	1	s_5	s_5	0	s_5
s_1	1	s_1	s_1	0	s_1	s_5	1	s_5	s_5	0	s_5
s_1	0	s_1	s_1	1	s_1	s_5	1	s_5	s_5	1	s_5
s_1	1	s_1	s_1	0	s_1	s_5	0	s_5	s_5	0	s_5
s_1	0	s_1	s_1	1	s_1	s_5	1	s_5	s_5	1	s_5
s_1	1	s_1	s_1	0	s_1	s_5	0	s_5	s_5	0	s_5
s_1	0	s_1	s_1	1	s_1	s_5	0	s_5	s_5	1	s_5
s_1	1	s_1	s_1	0	s_1	s_5	1	s_5	s_5	0	s_5
s_1	0	s_1	s_1	1	s_1	s_5	1	s_5	s_5	1	s_5
s_1	1	s_1	s_1	0	s_1	s_5	1	s_5	s_5	1	s_5
s_1	0	s_1	s_1	1	s_1	s_5	1	s_5	s_5	1	s_5
s_1	1	s_1	s_1	0	s_1	s_5	0	s_5	s_5	0	s_5

4. Conclusions

Any problem in the Computer Science can be virtualized in terms of the abstract machines viz., Finite Automata. Thus we have discussed the design of Automata that accepts the class of IP addresses. The proven validations in the results section states that the designed automata accept the corresponding class of IPV_4 Addresses.

References

[1] J.E.Hopcraft and Jeffery D.Ulman: Introduction to Automata Theory, Languages & Computation by– Narosa Publishing Company.
[2] Mishra &Chandra Sekharan: Theory of Computer Science PHI.
[3] Peter Linz: An Introduction to Formal Languages and Automata, 3e by– Narosa Publishing House.
[4] Sipser: Introduction to Theory of Computation, 2/e, Thomson.
[5] Daniel I.A. Cohen: Introduction to Computer Theory, John Wiley.
[6] Behrouz A.Forouzan, Data Communications and Networking, 3rd EditionTMH,2004
[7] William Stallings, Data and Computer Communications, 7th Edition, Pearson Education Inc., 2004.
[8] Martin JC. Introduction to languages and the theory of computation. 3rd edition Tata Mcgraw Hill; 2010.
[9] Saradhi Varma G.P, Tirupathi Rao B, Theory of Computation-Formal Languages and Automata Theory, Scitech Publications.
[10] P. Ezhilarasu, J. Prakash, N. Krishnaraj, D. Satheesh Kumar, K. Suresh Babu and C. Parthasarathy, A Novel Approach to Design the Finite Automata to Accept the Palindrome with the Three Input Characters, Indian Journal of Science and Technology, Vol 8(28), DOI: 10.17485/ijst/2015/v8i28/78189, October 2015.
[11] Ezhilarasu P, Krishnaraj N. Triple Substring Based Classification for Nondeterministic Finite Automata. IJAER. 2015; 10(59):177–82.
[12] Daniel I.A. Cohen, Introduction to Computer Theory, 2nd edition, Wiley Publishers 2007.
[13] S.N. Sivanandam, M. Janaki Meena, Theory of Computation, Published by I.K. International Publishing House Pvt. Ltd.
[14] MARTIN (Author), Introduction To Languages And The Theory Of Computation (), 3rd Edition
[15] S.R. Jena, S.K. Swain, Theory of Computation and Application-2017.

Authors' Profiles

Sri Ram Chandra. P has received his Bachelor's Degree in Computer science and Engineering from Andhra University, Master's Degree from GITAM University in the years 2010 and 2012 respectively. Currently he is a researcher in GITAM University with Theory of Computation and Cryptosystems Design & Cryptanalysis as research interests. At present he is working as Associate Professor in the department of Computer Science and Engineering, Godavari Institute of Engineering and Technology (A), Rajahmundry, Andhra Pradesh, INDIA. He is a member of CSI. He has published 03 research papers in reputed International journals. He has 6.4 Years of Teaching Experience.

K.S.Sravan is pursuing his Bachelor's Degree in Computer science and Engineering from Godavari Institute of Engineering and Technology (A), Rajahmundry, Andhra Pradesh, INDIA. He is in interested in Theory of Computations, Databases.

Srinivas Chakravarthy M has received his Bachelor's Degree, Master's Degree in Computer science and Engineering from JNTU-Kakinada in the years 2008 and 2011 respectively. Currently he is a researcher in GITAM University with Network Security as research interests. At present he is working as Associate Professor in the department of Computer Science and Engineering, Godavari Institute of Engineering and Technology (A), Rajahmundry, Andhra Pradesh, INDIA.

How to cite this paper: P.Sri Ram Chandra, K.S.Sravan, M.S.Chakravarthy,"A New Approach to the Design of a Finite Automaton that accepts Class of IPV_4 Addresses", International Journal of Mathematical Sciences and Computing(IJMSC), Vol.5, No.1, pp.65-79, 2019.DOI: 10.5815/ijmsc.2019.01.06