International Journal of Engineering Innovation & Research
Volume 1, Issue 2, ISSN : 2277 – 5668

IPv4/IPv6 Transition Using Singly Linked List

J. Hanumanthappa
Dept. of Studies in Computer Science, University of Mysore,
Manasagangothri, Mysore, India
hanums_j@yahoo. com

Dr. Manjaiah. D. H.
Dept. of Computer Science, Mangalore University,
Mangalagangothri, Mangalore, India
ylm321@yahoo. co. in

Abstract **- In the past few years, we have seen a rapid expansion in the field of wired networking due to the proliferation of inexpensive, widely available wireless devices. However current devices, applications and protocols are solely focussed on cellular or wireless local area networks(WLANS) not taking the account the great potential offered by wired networking. Wired local area networks make use of ethernet cables and network adapters. Numerous computers can be wired to one another by using an Ethernet crossover cable. Wired LANS also need vital devices like hubs, switches, or routers to aid further computers[2][3]. There are situations where IPv4/IPv6 transition is also possible using singly linked list. Un fortunately IPv4 and IPv6 are two important incompatible protocols. So this results in transition mechanism at the time of migration from IPv4 to IPv6 networks. Longest prefix matching (LPM) is a challenging innovative and creative topic because of the increasing routing table size, the increasing link speed and the increasing Internet traffic with decreasing traffic size. Due to the invention of IPv6 it requires the reconsideration of previous methods were highly essential to IPv4[1][3]. Hence the need is to introduce the first algorithm that we are aware of to employ BD-TTCS (Bi-Directional Talented Transmission Conversion System) based singly linked list is one of the technique specified in the literature to perform IPv4/IPv6 conversion whereas the Dual Stack transition technique is developed to perform IPv4/IPv6 transition for IPv6 dominant networks. In this paper we present a novel and simple approach to transition of IPv4/IPv6 transition based on singly linked list. This method splits the 128 bits IPv6 source address into 8 sections and the sections are reduced recursively through several tunable phases according to the trade off between lookup performance and memory consumption. The Proposed method uses singly linked method to speed up the splitting process of 128 bits IPv6 address into 8 chunks by means of BD-TTCS translator[1][2][3].**

Keywords **- IPv4, IPv6, IPv4/IPv6 Transition, Singly linked list.**

I. INTRODUCTION

The concept of singly linked list based migration of IPv4/IPv6 is a phenomenon that comes in divide and conquer based transition of IPv4/IPv6 mechanism. In level-1 the singly linked list concept is mainly used to split the entire 128 bits IPv6 source address into 8 sections of 16 bits and at level-2 the first node and second nodes of linked lists are fused into node-1. Fusing is a process of combining the split nodes that are obtained by scanning different parts of single large IPv6 address. Many a time it may not possible to send the entire complete 128 bits IPv6 address from Source to Destination in a single exposure. Several researchers have addressed different methods for obtaining the final entire 128 bits IPv6 address from its split sections[2][3].

Zhenqiang Li, Xiaohong Deng, Hongxiao Ma and Yan Ma described the usage of divide and conquer based scheme for IPv6 address longest prefix Matching. They have worked on divide and conquer based technique for 128 bits IPv6 Longest prefix matching (LPM). This algorithm splits an IPv6 address into 8 chunks of 16 bits each and the chunks are reduced recursively through several tunable phases according to the trade off between lookup performance and memory consumption.

Miguel A. Ruiz-Sanchez, Ernst W. Biersack, Walid Dabbous presented a survey of state-of-the art IP address lookup algorithms and compare their performance in terms of lookup speed, scalability and update overhead.

Srisuresh and Egevang in the year 2001 described a popular solution to the shortage of IPv4 addresses are Network Address Translation(NAT) which consists of hiding networks with private IPv4 addresses behind a NAT-enabled router with few public IPv4 addresses.

In their work Zeadally and Raicu in the year 2003 proposed the IPv6/IPv4 performance on Windows 2000 and Solaris 8. In their work they connected two identical personal computer's using a point-to-point connection. In order to calculate various performance issues namely Throughput, Round-trip time, CPU utilization, Socket-creation time and client–server interactions for both TCP and UDP. They used packets ranging from 64 to 1408 bytes. Their experimental results show that IPv6 for Solaris 8 outperform IPv6 for Windows 2000, while IPv4 outperform IPv6 for TCP and UDP.

In their work Zeadally and Raicu (2003) noted the IPv6/IPv4 performance on Windows 2000 (Microsoft IPv6 Technology Preview for Windows 2000) and Solaris 8. They connected two identical workstations using a point-to-point connection and reported results such as throughput, round-trip time, CPU utilization, socket-

creation time, and client–server interactions, for both TCP and UDP. They used packets ranging from 64 to 1408 bytes. Their experimental results show that IPv6 for Solaris 8 outperform IPv6 for Windows 2000, while IPv4 outperform IPv6 for TCP and UDP.

Zeadally et al. (2004) designed and calculated IPv6/IPv4 performance on Windows 2000, Solaris 8, and RedHat 7. 3. The authors experimentally measured throughput of TCP and UDP, latency, CPU utilization, and web-based performance characteristics. Mohamed et al. (2006) evaluated IPv6/IPv4 performance on Windows 2003, FreeBSD 4. 9 and Red Hat 9. They measured throughput, round-trip time, socket-creation time, TCP-connection time, and number of connections per second in three different test-beds. The first test-bed consists of a single computer and communication was limited to processes running in this computer using the loopback interface. In the second test-bed, two computers were connected through an Ethernet hub. The Ethernet hub was replaced by a router in the third test-bed. They used packets ranging from 1 byte up to the limits of an IP packet (which is typically around 65, 535 bytes).

Another solution to the problem of the shortage of public IPv4 addresses that faces the Internet consists to migrate to the new version of the Internet protocol(Davies, 2002;Deering and Hinden, 1998;Popoviciu et al. , 2006), called IPv6, or the coexistence between both protocols(Blanchet, 2006). IPv6 fixes a number of problems in IPv4, such as the limited number of available IPv4 addresses. IPv6 has a 128-bit address, while IPv4 has a 32-bit address.

Shiau et al. (2006) evaluated IPv6/IPv4 performance in two different scenarios. In the first scenario, they connected two identical computers using a point-to-point connection. In the second scenario, the two identical computers were each connected to a real large-scale network environment through a Cisco 3750GB switch, and a Cisco 7609 router. Fedora Core II was the operating system of the two computers. The authors reported results such as throughput, round-trip time, packet loss rate, for both TCP and UDP. None of these previous works compares the experimental results for TCP and UDP throughput to the maximum possible throughput.

Hence in this paper in order to achieve transition of IPv4/IPv6 a novel and simple approach is presented. Thus the crux of the work is determine the splitting of 128 bits IPv6 address into 8 chunks of 16 bits at level-1. In level-2 the node's 1 and 2 are merged into Node-1 and node's 3 and 4 are combined into Node-2 and node's 5 and 6 are merged into Node-3 and node's 7 and 8 are merged into Node-4. The advantage of this approach is that no exhaustive search is required. The proposed method has the worst case time complexity $o(n^3)$. The Fig. 1. depicts block diagram of the linear linked list based transition of IPv4/IPv6.

There is a great demand for developing an algorithm for transition of IPv4/IPv6 transition using linear linked list in BD-TTCS translator.

The paper is organized into 4 sections. Sections-2 clearly explains the proposed methodology to split 128 bits IPv6 address. The experimental results are reported in section-4. Finally the conclusion is given in section 5.

II. PROPOSED METHODOLOGY

The authors of this paper have proposed (Hanumanthappa. J. et al) a new technique to tackle the above mentioned problems. The above technique works for the various types of 128 bits IPv6 addresses. In this section novel approach is based on singly linked list for transition of IPv4 to IPv6 scenarios in BD-TTCS translator. A Singly linked list is a dynamic data structure in which all nodes are linked together in some sequential manner, hence it is also called linear linked list[1][2].

The linear linked list clearly specifies beginning and the end and it may grow or shrink depending upon of the operations made. Let us start the study of the singly linked list by first creating it. In C, a linked list is created using structures, pointers and dynamic memory allocation function like malloc. We consider head as an external pointer which helps in creating and accessing other nodes in the linked list. Consider the following structure definition and head creation[4].

```
struct node {int num; struct node * ptr;};
typedef struct node Node;
Node *head;
head=(Node*) malloc(sizeof(Node));
```

The proposed method is presented in three subsections. Sub sections 2. 1 shows how to split 128 bits IPv6 address into 8 chunks of 1 word size.

Algorithm-1:Splitting IPv6 address into 8 bytes using singly linked lists.

Input:128 bits of IPv6 Source address.

Output:128 bits IPv6 address.

Method:

Step-1: [The IPv6 Total address space should be represented in 32 Hexadecimal bits.

Split IPv6 address into 8 octet's and specify each octet address value through 1 to 8 using linear linked list].

Step-2:[Merge process at Level-1]

{Merge the IPv6 address value (Hexadecimal bits) in Octet-1 and Octet-2->Node-1 at level-2}.
{Combine IPv6 address value in Octet-3+Octet-4=Node-2}.
{Merge IPv6 Hexadecimal bits value in Octet-5+Octet-6=Node-3}.
[Merge IPv6 Hexadecimal address value of Octet-7+Octet-8=Node-4].
Step-3**:**[Merging Operation at Level-2]
Combine IPv6 hexadecimal values of Node-1 and Node-2 as Node-1 and Node-3 and Node-4 as
Node-2 at level-3.
Step-4:{Decoder operation in Level-3}
Merge IPv6 hexadecimal bits of Node-1 and Node-2 of level-3 as Node-1 at Level-4.
Method ends
Algorithm splitting IPv6 address into 8 bytes using singly linked list ends

2. 1. Linear Linked list based transition of IPv4/IPv6.

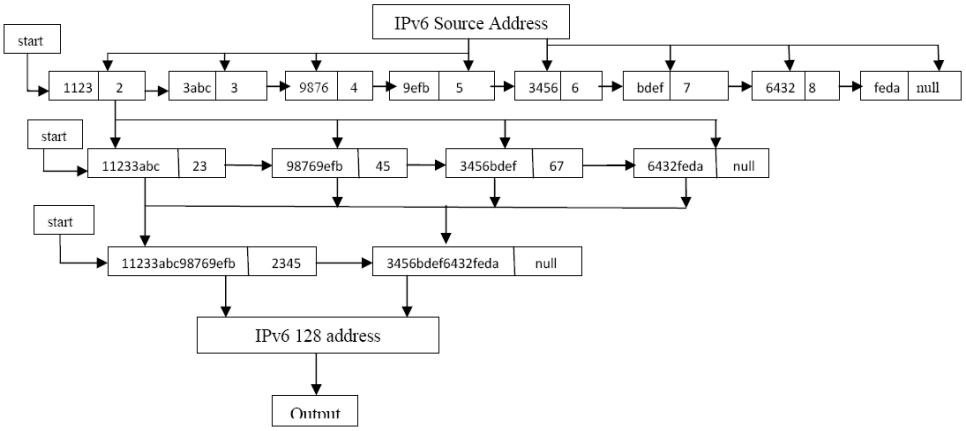

Fig. 1. Block diagram of the proposed work.

A Singly linked list(One-way list) is a linear collection of data elements called nodes, where each node is divided into two parts called information field and pointer field to the next node(link field). The information field holds the actual value to be stored and accessed in a list. The information field keep tracks of the actual value to be stored and accessed in a list. In other words, a linked list consists of series of structures which are not necessarily contiguous in memory. The pointer part (Link field) keeps track of the address which belongs to the next node. So the link field of the first node refers to the second node and the link field in the second node refers to the third node and so on. In other words it establishes a link to the next data item(node) in the linked list from the current node. We can create a linked list using two methods called iterative and recursive. The Algorithm-2 depicts how to create a linked list[4].

2. 2. IPv6 address generation using Linear linked lists.

This section clearly illustrates a simple approach to generate IPv6 address generation from singly linked lists. The algorithm to create IPv6 address using Linear linked list is depicted in Algorithm-2. The method creates a node and stores the address of this first node in start. So start will be pointing to the first node of the singly linked list. The step-2 of Algorithm-2 stores the address of the first node in another pointer CURRPTR so that Node-1 is now pointed by both start and CURRPTR. If the IPv6 address first consists of 4 hexadecimal decimal values like 1123 then 1123 is accepted from the keyboard and it is stored in info field of the first node (pointed by CURRPTR). When the choice is 'Y' and in order to create another node refer the procedures of steps-5. If the choice is 'N' and to make the link field of last node to NULL indicating the end of linked list refer step-6[4].
Algorithm-2:IPv6 address creation using Linear linked lists
Input: IPv6 address formation octet by octet.
Output: Formation of Linked lists
Method:
Step-1 : start=getnode();
Step-2 : CURRPTR=start;
Step-3 : INFO[CURRPTR]=ITEM;
Step-4 : If(choice ='Y') goto step 5 else goto step 6
Step-5 : i)NEWNODE=getnode();

ii)LINK[CURRPTR]=NEWNODE;
iii)CURRPTR=NEWNODE
iv)INFO[CURRPTR]=ITEM
v)go to step 4
Step-6 :LINK[CURRPTR]=NULL and Exit
Step-7:return
Method ends
Algorithm IPv6 address creation using linear linked list ends.

2. 3. The procedure to display 128 bits of IPv6 in hexadecimal notation.

Algorithm-3:IPv6 address display using singly linked list.
Input:IPv6 address in hexa decimal notation.
Output: IPv6 address in hexa decimal notation.
Method:
Step-1:[Check "LIST EMPTY"]
If (start==NULL) write ("LIST EMPTY") return;
Step-2 : set CURRPTR=start
Step-3:repeat step4 and step5 while (CURRPTR!=NULL)
Step-4:display the items one by one from the first node until CURRPTR becomes NULL]
method ends
Algorithm IPv6 address display using singly linked list ends.

2. 4. The procedure to traverse 128 bits of IPv6 using linked list.

Algorithm-4: IPv6 address traverse using singly linked list.
Input:IPv6 address in hexa decimal notation.
Output:IPv6 address in hexa decimal notation.
Method :
Step-1:[Is Linked list is empty?]
If (start=NULL) display ("Linked list is empty") exit)
Step-2:[Assign start value to CURRPTR]
CURRPTR=start
Step-3: Repeat while(CURRPTR!= NULL)
Process
INFO[CURRPTR] CURRPTR=LINK[CURRPTR]
Step-4: exit

2. 5. The procedure to search 128 bits of IPv6 using linked list.

Algorithm-4: IPv6 address search using singly linked list.
Input:IPv6 address in hexa decimal notation.
Output:IPv6 address in hexa decimal notation.
Method :
Step-1:set CURRPTR=start, LOC=NULL
Step-2:repeat step 3 while CURRPTR=NULL
Step-3: if(ITEM==INFO[CURRPTR])
then LOC=CURRPTR
and display ='search successful' exit
EXIT else
CURRPTR=LINK[CURRPTR]
Step-4 : if LOC=NULL
display 'Search is Unsuccessful'
Step-5 : EXIT

III. Performance Evaluation Metrics

Our primary performance metrics in this research work is RTT(Round Trip Time), Throughput. In order to calculate IPv4 and IPv6 RTT Ping command is one of an essential tool. The network performance is different for both TCP and UDP Protocols. We present the findings of the research in this section. The majority of the tests were done for a sufficiently long period of time and resulted in the swapping of 50, 000 packets to 1, 00, 00, 00 packets depending on the various size packets ranging from 0 bytes to 65538 bytes of packets sent and the corresponding test. We conducted an empirical calculations based on the following performance metrics: Round trip time(RTT), End-to-End delay(EED), Throughput for UDP traffic and TCP traffic, using the packet size ranging from 32 bytes to 1024 bytes in order to calculate the impact of small size packet as well as the larger size packet on the translation and address mapping processes used in BD-TTCS translator. All the performance metrics are computed using the NS-2 simulation with maximum bursty traffic where each packet arrival follows a Poisson distribution process with rate χ=2. In order to plot bar graph and Line Graph we used Matlab 7. 11. 0(R2010b).

IV. Simulations in NS2

NS2 (Network Simulator Version 2) developed by UC Berkeley is a kind of open-source free software simulation platform in allusion to network technology [1]. It's essentially a discrete event simulator[5][6][7]. There are 2 levels in the simulation of NS2 one is based on configuration and construction of Otcl, which can use some existing network elements to realize the simulation by writing the Otcl scripts without modifying NS2 the other is based on C++ and Otcl. Once the module resources needed do not exist, NS2 must be upgraded or modified to add the required network elements[7]. Under these circumstances, the split object model of NS2 is used to add a new C++ class and an Otcl class, and then program the Otcl scripts to implement the simulation. The basic architecture or main components of NS2 are shown in Fig. 2. NS2 now has become one of the first selected software to implement network simulation in the academic field[5][6].

Event Scheduler	*NS2*
Tclcl	Network component
Otcl	
Tcl/Tk	

Fig. 2. The main components of NS2 or Architecture of NS2.

4. 1. The basic components in NS2

NS2 simulation can be divided into two layers. At first we should analyze which layer is involved before the network simulation. One layer is based on OTcl programming. There are no needs to modify NS itself to implement simulation by use of existing network elements of NS, just to compile OTcl scripts. Another layer is the one based on C++ and OTcl programming. If there aren't required network elements in NS, it's needed to extend NS, adding required ones which also mean adding new C++ and OTcl class, then to compile OTcl script.

The simulations were performed using Network Simulator 2(Ns-2), particularly popular in the wired networking community. The traffic sources are CBR(continuous bit–rate). The source-destination pairs are spread randomly over the network. The wired network model uses 'random waypoint model' in a rectangular filed of 500m x 500m with 50 nodes. During the simulation, each node starts its journey from a random spot to a random chosen destination. Once the destination is reached, the node takes a rest period of time in second and another random destination is chosen after that pause time. This process repeats throughout the simulation, causing continuous changes in the topology of the underlying network. Different network scenario for different number of nodes and pause times are generated[5]. The various simulation parameters in NS2 for the transition of IPv4/IPv6 experiment using singly linked list is shown in Table-1[5][6].

Sl. No.	Parameter	Value
1	Simulator	Ns-2
2	Simulation time	200 s
3	Simulation area	500X500
4	Transmission range	250 m
5	Bandwidth	2Mbps
6	Traffic type	CBR
7	Data payload	Bytes/packet

Throughput : Ratio of the packets delivered to the total number of packets sent.

1. **Packet Delivery :** Packet Delivery Ratio in this simulation is defined as the ratio between the number of packets sent by constant bit sources (CBR) and numbers of packets received by CBR sink at destination.

Packet_Delivery%=$\sum_1^n CBR\ Sent/\sum_1^n CBR\ Received$ X100 ----------(1)

Minimum Delay: Minimum Time taken for the packets to reach the next node, **Maximum Delay**:Maximum Time taken for the packets to reach the next node, **Average End-to-End Delay**:Time taken for the packets to reach the destination

Avg_End_to_End_Delay=

$\sum_1^n CBR\ SentTime - CBR\ receivedTime/\sum_1^n CBR\ Received$ ----------(2)

Simulation Time:The time for which simulations will be run i. e. time between the starting of simulation and when the simulation ends, **Network size**: It determines the number of nodes and size of area that nodes are moving within. Network, size basically determines the connectivity. Very lesser nodes in the same area mean fewer neighbours to send request to, but also smaller probability of collision, **Number of Nodes:** This is constant during the simulation. We used 50 nodes for simulations, **Pause time**:Node will stop a "pause time" amount before moving to another destination point, **Jitter**:Jitter describes standard deviation of packet delay between all nodes.

Jitter(J) = $|D_{i+1} - D_i|$ ---------- (3)

Power Consumption: The total consumed energy divided by the number of delivered packet, **Average Packet Delay**:It is the sum of the times taken by the successful data packets to travel from their sources to destination divided by the total number of successful packet. The average packet delay is measured in seconds.

End-to-End (D)=$T_d - T_s$ ---------- (4)

Where T_d=Packet time received at destination node and T_s= Packet sent time at Source node, **Average Hop Count**:It is sum of the times taken by the successful data packets to travel from their sources to destination divided by the total number of successful packets. The average hop count is measured in number of hops, **Node Expiration time (NET):**It is the time for which a node has been alive before it must halt transmission due to battery reduction. The node expiration is plotted as number of nodes alive at a given time, for different point in time during the simulation, **Packet loss:**Packet loss is one which is defined as the network traffic fails to reach its destination in a timely manner. Most commonly the packet gets dropped before the destination can be reached.

Packet dropped/loss (P_d) =$P_s - P_r$ ----------(5)

Where P_s is the amount of packet sent at Source and P_r is the amount of packets received at Destination,

Where D_{i+1} are delay of i^{th}+1 packet and D_i is the delay of i^{th} communication packet.

*4. 2. Analysis of Simulation Results***.**

In this section we validate our proposed algorithm with the help of discrete event simulation. The National Science Foundation(NSF) network is considered for our simulation study. The topologies shown in Figs. 3 consist of bidirectional links, each carrying data at a rate of 10 Gbits/s. We assume that there is no wavelength conversion and regeneration capability for the network. Burst arrivals

follow a Poisson process with an arrival rate of λ bursts/s. The length of the burst is exponentially distributed with the expected service time of 1/μ s. The network load is defined as λ /μ. Links in Figs. 3 benefit from in-line erbium-doped fiber amplifiers(EDFAs) placed 70 km apart. The calculation of the noise factor is based on linear impairments such as attenuation, mux/demux loss, tap loss. and amplified spontaneous emission noise. There are no optical buffers, and hence the burst that finds the channel occupied will be dropped or lost. The reliability factor of the link indicates that the reliability is affected by damage caused by faults, fiber cuts, and catastrophic effects

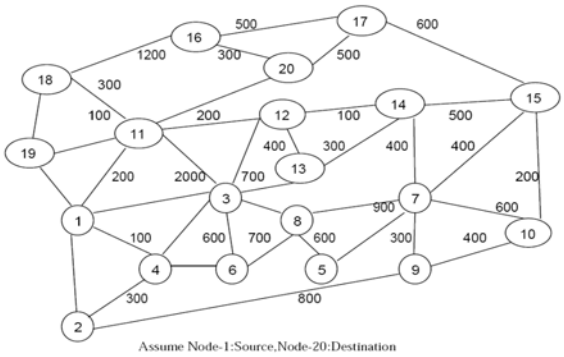

Fig. 3. Network topology of NSF network consisting of 20 nodes with 35 bidirectional fibre optic links

V. SIMULATION RESULTS

Fig. 4. shows the part of our simulation result over the simulation time 0 to 60 seconds for the Packet delivery Fraction ratio versus Pause time when the size of the packet varies from 0 to 65536 for 50 nodes with 10 Sources.

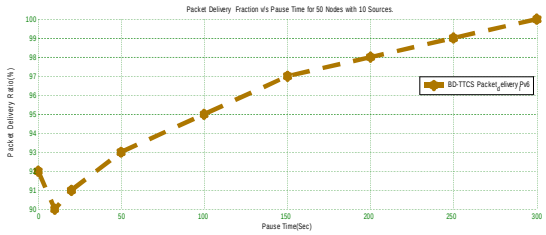

Fig. 4. Packet delivery Fraction ratio versus Pause time for 50 nodes with 10 Sources

Fig. 5. shows the part of our simulation result over the simulation time 0 to 60 seconds for the End to End delay versus Pause time when the size of the packet varies from 0 to 65536 for 50 nodes with 10 Sources.

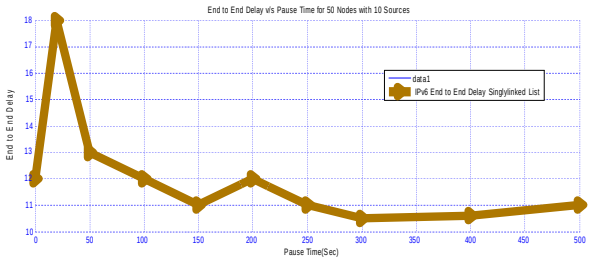

Fig. 5. End to End delay versus Pause time for 50 nodes with 10 Sources

5. 1. Analysis of Graph of Throughput versus number of nodes.

Fig. 6. and Fig. 7. shows the part of our simulation result which shows Packet delivery Fraction ratio versus Pause time for 50 nodes with 10 Sources and End to End delay versus Pause time for 50 nodes with 10 Sources.

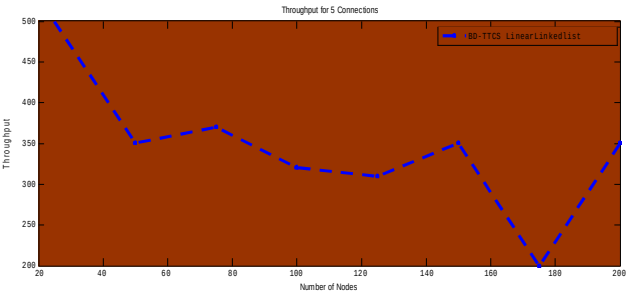

Fig. 6. Graph of Number of Nodes v/s Throughput for 5 Connections in IPv6 with Singly linked List

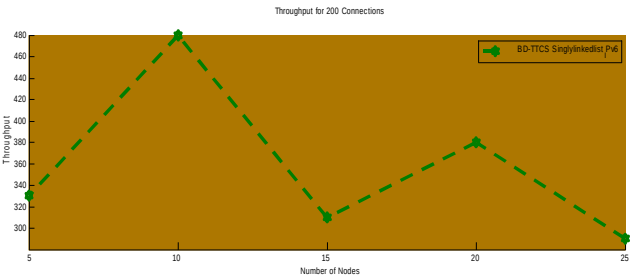

Fig. 7. Graph of Number of Nodes v/s Throughput for 200 Connections in IPv6 with Singly linked List

ACKNOWLEDGEMENTS

The authors would like to thank UGC for providing funding for this project with an entitled “Design Tool of IPv6 Mobility for 4G-Networks”, under eleventh plan of Major Research Project scheme(Ref. No F. No 36-167/2008(SR) dated . 26. 03. 2009).

REFERENCES

[1] Hanumanthappa. J. and Manjaiah. D. H. , A Mathematical Model To Realize The Parallel Computing Based Diminution Tree With List Ranking Based BD-TTCS As a New IPv4/IPv6 Transition Mechanism Proceedings of the International Conference on Advanced Computing, Networking and Security (Adcons-2011), NITK, Surathkal, Karnataka, India, December-16-18, 2011, pp 579-585.

[2] Hanumanthappa. J. and Manjaiah. D. H. , A New Scheme for IPv6 BD-TTCS Translator Proceedings of the 8th International Conference on Distributed Computing and Information Technology(ICDCIT-2012), KIIT University, Bhubaneswar, Odisha, India, 02-04, February, pp 106-116.

[3] Hanumanthappa. J. , Dr. Manjaiah. D. H, A Simulation study on the performance of Divide-and-Conquer based IPv6 Address LPR in BD-SIIT IPv4/IPv6 transition using a Novel Reduced Segment Table(RST) algorithm in BD-SIIT Translator Proceedings of the International Conference on Computer's and Computing(ICCC'11), Lanzarote, Canary Islands, Spain, May, 27-29, 2011, pp 146-152.

[4] Data Structures with "C", P. B. Kotur.

[5] NS-2 network simulator http://www. isi. edu/nsnam/ns.

[6] Network Simulator-2(NS-2) http://mohit. ueuo. com/NS-2. html.

[7] Marc Greis'Tutorial for the UCB/LBNL/VINT Network Simulator- ns. http://www. isi. edu/nsnam/ns/tutorial/

[8] Larry L. Peterson and Bruce s. Davie. Computer Networks-A Systems Approach. Edition-3 Morgan Kaufmann Publishers.

[9] Sibeyn, J. F. , F. Guillaume, T. Seidel, 'Practical Parallel List ranking', Proc. 4th Symposium on Solving Irregularly Structured Problems in Parallel, LNCS 1253, pp. 25-36, Springer- verlag, 1997.

[10] P. Mockapetris, *Domain names—Implementation and Specification*, Nov. 1987, RFC 1035.

[11] Droms, R. , Bound, J. , Volz, B. , Lemon, T. , Perkins, C. , and M. Carney, "Dynamic Host Configuration Protocol for IPv6 (DHCPv6)", RFC3315, July 2003.

[12] Classic Data Structures, D. Samanta, Eastern Economy Edition, Prentice –Hall of India.

[13] Theory and Problems of Data Structures, Seymour Lipschutz, International Editions(Schaum's Outline Series).

[14] H. Afifi, and L. Toutain;"Methods for IPv4-IPv6 Transition", IEEE, 1999.

[15] E. Nordmark, "Stateless IP/ICMP Translation Algorithm (SIIT)", RFC 2765, 2000.

[16] Jivika Govil, Jivesh Govil, Navkeerat Kaur, Harkeerat Kaur, An examination of IPv4 and IPv6 Networks:constraints and various transition mechanisms.

[17] Hennessy, John L. and David A. Patterson (2002). Computer Architecture:A Quantitative Approach. 3rd edition, Morgan Kaufmann, p. 43. ISBN 1-55860-724-2.

[18] Kurose. J. & Ross. K. (2005) Computer Networking:A top-down approach featuring the Internet. 3rd ed, (Addison Wesley).

[19] Atul Kahate, "Cryptography and Network Security", Tata McGraw-Hill, 2003, pp-8-10.

[20] Ioan R, Sherali. Z. 2003. Evaluating IPv4 to IPv6 Transition mechanism. IEEE, West Lafayette, USA, v (1):1091–1098.

[21] Savola P. Security implications and considerations of internet protocol version 6(IPv6), October 2005.

[22] Jivika Govil, Jivesh Govil, Navkeerat Kaur, Harkeerat Kaur, "On the Investigation of Transactional and Interoperability Issues between IPv4 and IPv6.

[23] N. Oliver and V. Oliver;"Computer Networks Principle, Technologies and Protocols for Network Design", John Wiley and Sons, England, 2006.

[24] E. Nordmark, and R. Gilligan;"Basic Transition Mechanisms for IPv6 Hosts and Routers", RFC 4213, October 2005.

[25] Jun. Bi, J. Wu, and X. Leng, "IPv4/IPv6 Transition Technologies and Univer6 Architecture", International Journal of Computer Science and Network Security, VOL. 7 No. 1, January 2007.

[26] E. Park, J. Lee, and B. Choe, "An IPv4-to-IPv6 dual stack transition mechanism supporting transparent connections between IPv6 hosts and IPv4 hosts in integrated IPv6/IPv4 network", IEEE International Conference on, Vol. 2, 2004.

AUTHOR'S PROFILE

Mr. Hanumanthappa. J. was born on 06th December, 1975 at Harihar, Davanagere (Dist) (Karnataka). He had a consistently brilliant academic record and completed his B. E. (CS & E) from University B. D. T College of Engineering, Davanagere Karnataka (S), India (C), from the Kuvempu University, at Shankarghatta, Shimoga in 1998 and M. Tech (CS&E) from NITK Surathkal, Karnataka(S), India (C) in 2003 obtaining 1st Class and he joined the Dept of Studies in Computer Science, University of Mysore, Manasagangothri, Mysore on 04th June 2004. He has been an eminent teacher and researcher in Computer Science and Engineering having made an excellent contribution towards Teaching and Research for nearly 11 years in so many engineering colleges in Davanagere, Bangalore and DoS in CS at Mysore. Currently he is Teacher Fellow at the Dept of P. G Studies and Research in Computer Science, from Mangalore University for his Ph. D in Computer Science under the supervision of Dr. Manjaiah. D. H on an area of Wired Networks and on the topic entitled "Investigations into the Design, Performance and Evaluation of a Novel IPv4/IPv6 Transition Scenarios and also presently he is working as an Asst. Professor at DoS in Computer Science, Manasagangotri, Mysore. He has also co-authored more than 45 innovative technical papers out of which 32 in International Conferences, 9 in National Conferences and 4 papers in refereed International Journals. His prime area of research has been Computer networking, Protocol design, Network Protocols and Algorithms, Protocol performance issues, etc.

He is an active researcher in the areas like Wire/Wireless Networking, Computer Networks, Ad-hoc Networks, Mobile Adhoc Networks, Sensor Networks etc. He is a life member of Institution of Engineer's(IEI) and Indian unit of Indian Society for Technical Education(ISTE). He has been serving as an Associate Editor, Guest editor, co-editor, Session chair in International/National Conferences and Programme committee member and Reviewer of Many International Journals and conferences. He has supervised large number of students at M. Sc, MCA, and M. Tech level. He has also provided services to advisory boards in of several International and National level conferences. Thus Mr. **Hanumanthappa. J.** in his long, active and innovative academic career has not only dedicated and contributed significantly to the area of Protocol Performance issues but has, in fact, been instrumental in launching a crusade for introduction of value education and promotion of holistic approach in the Indian technical education system. He also has a unique distinction of carrying out innovative work towards the Integration of Science, Technology and Human values and introducing a holistic approach to technical education as well as Technology development.

Dr. MANJAIAH D. H. is currently Professor and Chairman of the Dept. of Computer Science. , Mangalore University, and Mangalore. He is also the BoE and BoS Member of all Universities of Karnataka and other reputed universities in India. He received **Ph.D. degree** from University of Mangalore, **M. Tech. from NITK, Surathkal** and **B. E. from Mysore University.** Dr. Manjaiah D. H has more than 15-years extensive academic, Industry and Research experience. He has worked at many technical bodies like **CSI[AM IND 00002429], ISTE[LM-24985], ACS, IAENG, WASET, IACSIT and ISOC**.

He has authored more than-50 research papers in International / National reputed journals and conferences. He is the recipient of the several talks for his area of interest in many public occasions. He had written Kannada text book, with an entitled, **"COMPUTER**

PARICHAYA", for the benefits of all teaching and Students Community of Karnataka. Dr. Manjaiah D. H 's areas interest are **Advanced Computer Networking, Mobile/Wireless Communication, Wireless Sensor Networks, Operations Research, E-commerce, Internet Technology** and **Web Programming**. He is the expert committee member of various technical bodies like **AICTE**, various **Technical Institutions** and **Universities** in INDIA. He had been actively involving in chairing technical sessions of various International & National Conference and reviewer of the Journals.

He is working with Major Research project on "**Design Tool for IPv6 Mobility for 4G–Networks**", around **Rs. 12 lakhs worth funded by UGC, New Delhi from year 2009-20 12**. He is recognized as a Ph. D. guide in Computer Science at Mangalore University, Mangalore and currently five students are doing their Ph. D. , under the guidance of him. He is recognized as advisory editorial board member of the International Journal of Advanced Computing [IJAC], International Journal of Computer Science and Application [IJCSA], and Journal of Intelligent System Research and Journal of Computing. He visited most of the countries in the actively involving in chairing technical sessions of various International & National Conference and reviewer of the Journals.

He is working with Major Research project on "**Design Tool for IPv6 Mobility for 4G–Networks**", around **Rs. 12 lakhs worth funded by UGC, New Delhi from year 2009-2012**. He is recognized as a Ph. D. guide in Computer Science at Mangalore University, Mangalore and currently five students are doing their Ph. D. , under the guidance of him. He is recognized as advisory editorial board member of the International Journal of Advanced Computing [IJAC], International Journal of Computer Science and Application [IJCSA], and Journal of Intelligent System Research and Journal of Computing. He visited most of the countries in the World.