International Journal of Emerging Technology and Advanced Engineering
Website: www.ijetae.com (ISSN 2250-2459, ISO 9001:2008 Certified Journal, Volume 7, Issue 3, March 2017)

Method Implemented for Simulating Networks with MDBR Algorithm

G. Raghu Ram[1], S. Anuradha[2], A. Subramanyam[3]

[1]*Research scholar, Dept. of Computer Science, Rayalaseema University, Kurnool, A.P, India.*
[2]*Post Doctoral Fellow UGC New Delhi, Vardhaman College of Engg. Kacharam, Shamshabad, Hyderabad, Telengana, India*
[3]*HOD, Dean & Professor CSE Dept., AITS, Rajampet,YSR Dt. Andhra Pradesh, India.*

***Abstract*— This paper covers the method used to implement for simulating Networks with MDBR(Modified Depth Breadth Routing) algorithm which is modified version of DBR algorithm which focus was on routing method where as MDBR is enhanced with load balancing to handle the congestion while routing.**

In the previous papers published on MDBR covered algorithm generation, implementation, method of node balancing ect., Now focus is on the simulation of networks using MDBR.

***Keywords*—Events, Event-driven, Priority queues, Packet movements.**

I. Introduction

The routing algorithm is very prominent in searching and selecting the proper pathway for data transmission from source to destination in modern communication systems.

As there are huge number of acceptable algorithms to route and locate the minimum feasible path which increases the throughput of the network. Proposed MDBR which routes to locate shortest path with node balancing on available nodes in the selected shortest path. In this paper we are going to explain the simulation method implemented.

To facilitate an empirical study of the behavior of mentioned routing algorithms. In the proposed network model, it is necessary to simulate the movement of packets over time in the model. To this end we have designed and implemented an event driven simulation scheme. The simulation is an adaptation of the event driven simulation scheme proposed in [Weiss 1998], where a priority-queue ensures that events are executed in order.

The proposed paper is represented in six sections with brief idea about network routing and the concept of Event location problem in Section II. Section III simulation of events, in section IV Empirical observations and analysis, In section V Experimental analysis and in Section VI conclusion.

II. Concept Of Event Location

An event-driven simulation only needs to contain a number of events with a specified time, and an event-holder, which handles the scheduling of these events. We have a system, where the defined events are spawned by other events. This is simply because a packet travels in a router-wire-router-wire-… cycle. The complexity in the system lies in knowing when a packet is ready for a switching operation or a sending operation (at a router or wire respectively), because of the buffering in the input- and output queues. Furthermore, in the formal case we had to define specific functions that handles the special cases, where another packet is already being processed (*timeSinceLastRoutingStarted*() and *timeToBufferEmpty*()). When implementing this in the event-driven simulation, we came upon an idea that could simplify this slightly. The basic idea is to keep track of when the *next event* is allowed to happen on a router or wire. The router or wire simply holds a time value that denotes the *next* time a switching/sending is not in progress. This means that an event can easily calculate the time it can schedule a next event.

This will be always:

nextEventTime(*router/wire*) + *durationOfThisEvent* , if *now*() < *nextEventTime*(*router/wire*)

now() + *durationOfThisEvent* , if *now*() ≥ *nextEventTime*(*router/wire*)

If the *nextEventTime* is sometime in the future, we must schedule the *next* event by the time-value in *nextEventTime*. Else, the router/wire is ready and the event can be scheduled by the present time: *now*(). The *durationOfThisEvent* denotes the duration of the current event, which of course must finish before the next event in this location can begin. It is now required of an event, that it updates the *nextEventTime* of the router or wire.

International Journal of Emerging Technology and Advanced Engineering
Website: www.ijetae.com (ISSN 2250-2459, ISO 9001:2008 Certified Journal, Volume 7, Issue 3, March 2017)

The update-value will generally be the same time that the new event was scheduled for; this is exactly the time that the router has finished switching or the wire is finished sending We have chosen to name this concept *event location*, as it refers to the fact that we utilize that events happen on locations in this simulation.

III. Simulation Of Events

We will now define the events that can occur in this event-driven simulation. The goal here is to simulate routing in our network model, a task requiring several steps that has already been discussed .We will briefly outline the events we have defined, their duration and purpose. The events and their properties should follow naturally from the description of routing in the network model and the discussion of packet loss.

The packet creation is initiated by an *input queue event.* This should be interpreted as if an external entity is responsible for creating the packet and has sent it to the initial router in our model. This mean that all new packets must wait in the input queues, and they risk being lost if the queue capacity is exceeded.

A. Input queue event

This event occurs whenever a packet arrives at an input queue on a router. The packet is lost if the input queue capacity is exceeded when it is attempted to insert the packet in the input queue.

If the queue is not full, an input queue event creates a *routing event*. This event is scheduled to occur when the packet has waited in the input queue and has been processed by the switching fabric. The time at which the *routing event* occurs is given by:

RoutingCompleteEventTime(*ri*,*t*) = *nextEventTime*(*ri*,*t*)+ *speed*(*ri*) , if the input queue is *not* empty

RoutingCompleteEventTime(*ri*,*t*) = *now*() + *speed*(*ri*), if the input queue is empty

(- where ri is the router where the event is located in the network)

B. Routing event

This event occurs when a packet has been processed in the switching fabric of a router. The length of the input queue on the router is decremented – note that this happens implicitly because the queues are only simulated with the help of the nextEventTime. If the packet has not arrived at its destination, it calculates the next router for the packet by using a routing algorithm. If the selected output queue is not full, its length is incremented (implicitly by way of nextEventTime).

A start-sending event is scheduled to occur, when the packet has waited in the output queue. The new event time is calculated by:

StartSendingEventTime(ri,rj,t) = nextEventTime(ri,rj,t),

if the output queue is not empty

StartSendingEventTime(ri,rj,t) = now(),

if the output queue is empty

(- where (ri,rj) is the wire connecting the event location router ri and the destination router rj in the network)

Note that we do not include the time to be sent over the wire in this event. This is because we have elected to include a transmission buffer between the output queue and the wire. As the internal transfer speeds are zero, it should be possible for a packet to be moved directly to the transmission buffer, if the output queue is empty.

C. Start Sending Event

This event happens when a packet is in the transmission buffer, and ready to be transferred over a wire. Implicitly it decrements the length of the output queue in which the packet was located to indicate the move to the transmission buffer. It then creates an input queue event to indicate that the packet arrives at the destination after being sent over the wire connecting the current router with the selected target. The calculation of the time for the new input queue event is simply:

InputQueueEventTime = *t* + *d*(*ri*, *rj*)

(- where ri is the router where the events happen, and rj is the next router selected by the routing calculation. t is the time when the start sending event happens).

D. Ant packets in the simulation Networks

As DBR and MDBR algorithms uses Ant colony optimization (ACO) methodology as their basic simulation while implementing.

It is clear from the above that specified events occur in a circular fashion with one creating the next until the packet arrives at its destination. Note that this is similar to the movement of ants in the ACO metaheuristic. Here the ants move from vertex to vertex until they have reached their success criteria or have died. In our ant based routing algorithm ants move as packets (aptly named ant-packets) and perform stochastic state transitions and lay trail as ants in the ACO metaheuristic. The main difference between the network simulation and the ACO metaheuristic is that ants move concurrently instead on sequentially.

International Journal of Emerging Technology and Advanced Engineering
Website: www.ijetae.com (ISSN 2250-2459, ISO 9001:2008 Certified Journal, Volume 7, Issue 3, March 2017)

IV. Empirical Observations And Analysis

In the following report the results and tendencies were found while developing and testing the routing algorithm on a few limited simulated networks.

Here some restrictive simplifications were made about the network in the experiments:

1. The packets in the network are created at a constant interval. There are no peak load situations with a sudden increase of traffic, where more packets are created in the network as a whole.
2. There is always only one router sending and only one router receiving packets. There is no other traffic on the network.

The results of our tests should be seen as a demonstration of the procedures could follow in a more thorough and systematical tests. The experiments use the set of fittings for the algorithm that are defined below. These fittings have been found to be effective through "trial-and-error" methods during the study

- α: 1 – the relative importance of trail-values.
- β: 3 – the relative importance of local heuristic-values.
- p_r: 0.02 – the probability of a packet being routed only using a restricted random step (as defined in [*Restricted random walk*].
- s_τ: 10 – the scale of the trail deposited, this value must be smaller than or equal to the minimum wire delay in the network.
- n_τ: 0.3 – the trail removed from a routing table entry on router, when an ant-packet is lost on the next router in the path of the packets.
- b_τ: 0.1 – the parameter used to for setting a lower and upper bound on trail-values In the routing tables, This means that the allowed range for trail-values is [0.1;0.9]

The quality of routing in the network is measured by the average distance traveled by all packets. This parameter (together with a number of relevant data concerning the simulation) are collected or calculated during a simulation and stored as output of the simulation. Furthermore the amount of dead packets are measured at fixed intervals and stored in another output-file.

For every run of a simulation it generated a picture of the network with a graphical representation of the number of ant-packets that has traversed each wire.

The network definition file and the simulation setup file are copied to the aforementioned directory. The network definition file is the file that defines the topology of the network (number of routers and wires, and the attributes of these).

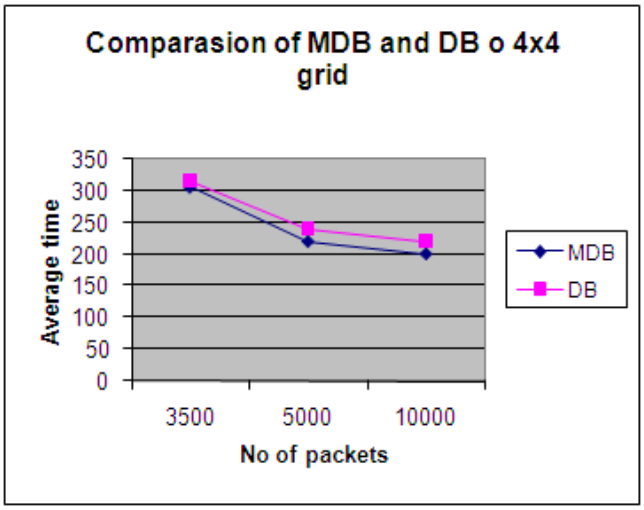
The simulation setup file defines the routing algorithm to use, the parameters that the algorithm will use, the duration, if and when to collect statistical data, the packet creation interval, which routers create the packets, and which routers receive the packets.

V. Experimental Analysis

Performance Analysis of Modified Depth-Breadth routing algorithm on different topologies with different network load

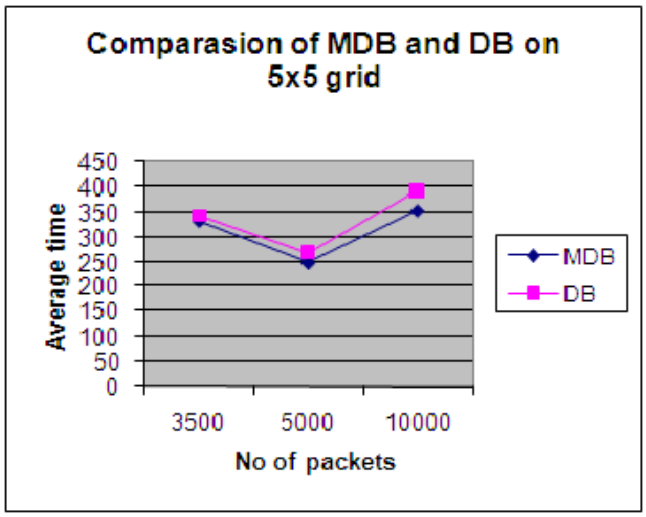
In this section 10 different test runs were conducted on three topologies namely 4x4 with 16 nodes, 5x5 with 25 nodes, 10x10 with 100 nodes of different packet loads of 1000, 3500,5000 and 10000 and the results were displayed in the following graphs.

The study effects on increase in packet load which increases the average time for the packet transmission in milliseconds on each topology this is obvious because of the increase in load and in number of nodes in the network.

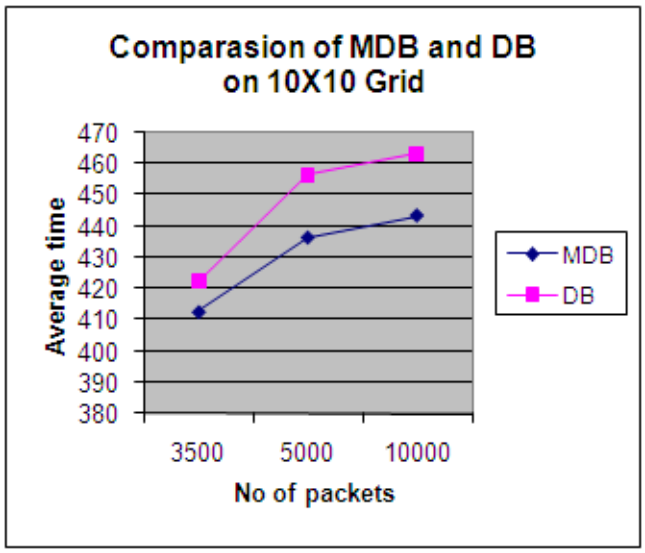


Graph1 Comparing results of Number of packets Vs Average time for packet in 4X4 grid using MDB and DB algorithms.

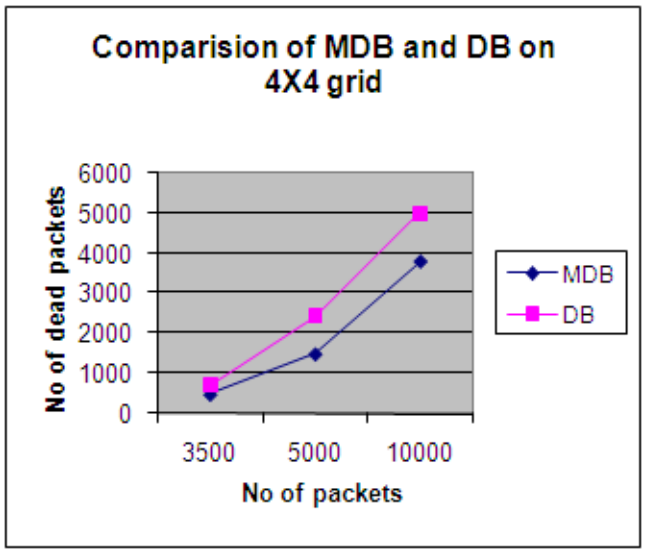
International Journal of Emerging Technology and Advanced Engineering
Website: www.ijetae.com (ISSN 2250-2459, ISO 9001:2008 Certified Journal, Volume 7, Issue 3, March 2017)



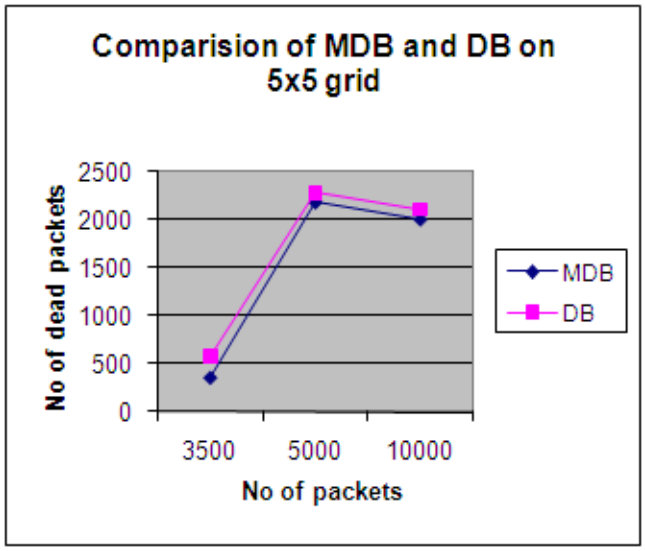
Graph.2 Comparing results of Number of packets Vs Average time for packet in 5X5 grid using MDB and DB algorithms.



Graph.3 Comparison of MDB and bench mark algorithms for No. of packets Vs Average time for packet in 10X10 grid.

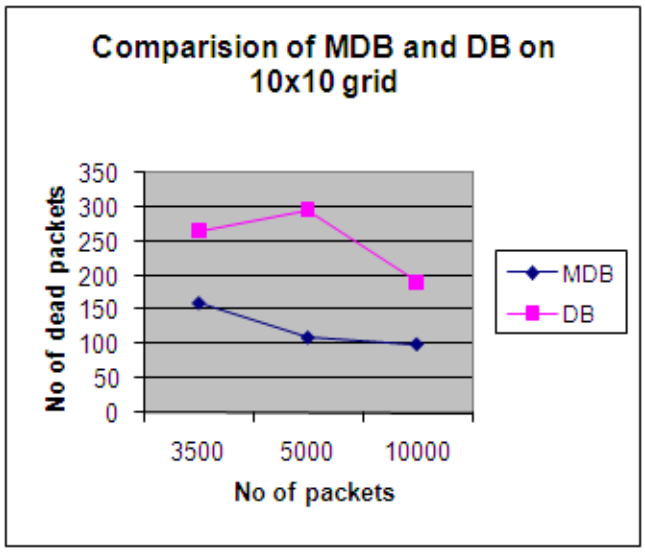


Graph.4 Comparison of MDB and bench mark algorithms for No. of packets Vs No. of dead packets in 4X4 grid.



Graph.5 Comparison of MDB and bench mark algorithms for No. of packets Vs No. of dead packets in 5X5 grid.

International Journal of Emerging Technology and Advanced Engineering
Website: www.ijetae.com (ISSN 2250-2459, ISO 9001:2008 Certified Journal, Volume 7, Issue 3, March 2017)



Graph.6 Comparison of MDB and bench mark algorithms for No. of packets Vs No.of dead packets in 10X10 grid.

VI. CONCLUSION

Different experiments were conducted by slightly varying the parameter values of MDB Routing algorithm on different sized topologies with variation of network load and compared with DB routing algorithm as Benchmark.

Here the experiments were conducted on 4x4, 5x5 and 10x10 grid topologies consisting of 16 nodes,25 nodes and 100 nodes respectively. This experiment started with the initial network load as 3500 and later with 5000 and 10000 packets.

In **DBR** algorithm the average time for packets slightly increasing with increase on the network load and the dead packet count is also slightly higher than MDB which deteriorates the performance. This decrease in the performance is because of lack of load balancing at each node in the algorithm. The average packet time in **MDBR** algorithm decreases as there is an increase in the network load due to which the performance is high and no of dead packets is also decreased due to the node balancing concept in the MDB routing algorithm. On the other hand it simultaneously improves the throughput with decrease in dead packets and decrease in average time of the packet traversal. This comparison concludes that the MDB routing algorithm is more effective compared to DBR algorithm with respect to packet transmission in the network communication.

Acknowledgement

The Part of this work is supported by Post Doctoral Fellow Awardee Anuradha Sanike (No. F.15-1/2014-15/PDFWM-2014-15-GE-AND-26672) by University Grants Commission (UGC), New Delhi, India.

REFERENCES

[1] Marco Dorigo & Thomas Stutzle - "Ant Colony Optimization" - MIT press edition, year: 2004.

[2] Routing - [Online Article] - Accessed on: 03.06.2011. - Link: http://en.wikipedia.org/wiki/Routing

[3] Dijkstra's algorithm - [Online Article] - Accessed on: 03.06.2011. - Link:http://en.wikipedia.org/wiki/Dijkstra%27s_algorithm

[4] Prim's algorithm - [Online article] - Accessed on: 02.06.2011. IEEE

[5] Mcgraw-Hill's Advanced Topics in Computer Science Series, New Ideas in Optimization.

[6] Gianni Di Caro - "Ant colony optimization and its application to Adaptive routing in telecommunication networks" - year: 2004.

[7] M. Dorigo, V. Maniezzo & A. Colorni, 1996. "Ant System: Optimization by a Colony of Cooperating Agents", IEEE

[8] S.Anuradha, G.RaghuRam, K.E.Sreenivasa murthy, " Evolution Of New Searching Technique In Unicast Routing Algorithm Of Data Networks**"** Praise Worthy Prize's International Review on Computers and Software(IRECOS) Voume.4,n°3 ,May 2009,pages(321-330)**.**

[9] S.Anuradha, G.Raghu Ram, "Dynamic Congestion Control using MDB-Routing Algorithm " Springer's Institute of Engineers India-B(IEI-B)series Journal May 2014.

[10] Gianni Di Caro, Marco Dorigo, AntNet: distributed stigmergetic control for communication networks, Journal of Artificial Intelligence Research 9 (1998) 317–365.

[11] M. Welzl, W. Eddy, Congestion Control in the RFC Series, University of Innsbruck, October 30, 2012

[12] S. Anuradha, "New Routing Technique to improve transmission Speed of Data Packets in Point to Point Networks" ICGST Inter National Journal on computer Networks and Internet Research(CNIR) Volume 8,Issue II January 2009,Pages(1..10)

[13] Elke Michlmayr, Arno Pany, Sabine Graf: "Applying Ant-based Multi- Agent Systems to Query Routing in Distributed Environments," 3rd IEEE Conference On Intelligent Systems (IEEE IS06), London, UK, September 2006

[14] Elke Michlmayr: "Ant Algorithms for Search in Unstructured Peer-to- PeerNetworks," Ph.D.Workshop, 22nd International Conference on Data Engineering (ICDE 2006), Atlanta, Georgia, USA, April 2006

[15] S. Anuradha, , K.E. Sreenivasa murthy, B. Gurunath Reddy, "Fast Transfer of Packets Through DB Routing Using Ant Colony Optimization" Praise Worthy Prize's International Review on Computers and Software(IRECOS) Vol.3 N.4 July 2008, Pages(349..355)

[16] Lilia Rejeb and Zahia Guessoum. The Exploration-Exploitation Dilemma for Adaptive Agents. In Proceedings of the 5th European Workshop on Adaptive Agents and Multi-Agent Systems (AAMAS05), March 2005

[17] S.Anuradha, G.RaghuRam, K.E.Sreenivasa murthy, V.Raghunath Reddy, B.Satya narayana "Genesis of DB Routing Algorithm in Unicasting Networks" ICGST Inter National Journal on computer Networks and Internet Research(**CNIR**) Volume 9, Issue I, July 2009, Pages (47.. 55)

[18] St ¨ utzle and Marco Dorigo. A short convergence proof for a class of ant colony optimization algorithms. IEEE Transactions on Evolutionary Computation, 6(4):358–365, August 2002

[19] S.Anuradha, G.RaghuRam, K.E.Sreenivasa murthy, V.Raghunath Reddy, B.Satya narayana " Evolution Of New Searching Technique In Unicast Routing Algorithm Of Data Networks" Praise Worthy Prize's International Review on Computers and Software(IRECOS) Voume.4,n°3 ,May 2009,pages(321-330)

[20] Dorigo, Marco and Stützle, Thomas. (2004) Ant Colony Optimization, Cambridge, MA: The MIT Press

[21] W. Chung, ―Congestion control for streaming media,‖ Ph. D. dissertation, Polytechnic Inst., Worcester, 2015.

[22] S. Floyd, Ed., "Metrics for the Evaluation of Congestion Control Mechanisms", March 2008.

[23] De Marco, F. Postiglione, M. Longo, ―Run-time adjusted congestion control for multimedia: experimental results‖ , Journal of Interconnection Networks (JOIN), vol. 5, no. 3, pp. 249-266, 2004.

[24] A. Kuzmanovic, A. Mondal, S. Floyd and K. Ramakrishnan "Adding Explicit Congestion Notification (ECN) Capability to TCP's SYN/ACK Packets" AT&T Labs Research, June 2014.