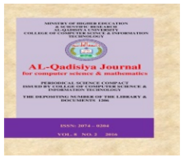
Journal of Al-Qadisiyah for Computer Science and Mathematics **Vol. 14(4) 2022 , pp Comp. 12–25**

University Of AL-Qadisiyah

Available online at www.qu.edu.iq/journalcm

JOURNAL OF AL-QADISIYAH FOR COMPUTER SCIENCE AND MATHEMATICS

ISSN:2521-3504(online) ISSN:2074-0204(print)



Traffic Load Balancing based on Congestion Avoidance (TLBCA) in high-speed computer networks

Ahmed Malik Khudhair[1], Khulood A. Nassar[2]

[1]Department of Computer Information System, College of Computer Science and Information Technology, Basra, Iraq
Email: itpg.ahmed.malik@uobasrah.edu.iq
[2]Department of Computer Information System, College of Computer Science and Information Technology, Basra, Iraq
Email: khulood.nassar@uobasrah.edu.iq

ARTICLE INFO

Article history:
Received: 10 /09/2022
Rrevised form: 15/10/2022
Accepted : 20 /10/2022
Available online: 01/12/2022

Keywords:

Congetion Avoidance

Congestion Control

Computer Netwoeks

Close Loop

ABSTRACT

Congestion, and how to manage it or prevent its occurrence, is one of the most crucial and cutting-edge subjects in the world of networks since it has a significant influence on the computer network and the quality of service (QoS). Network efficiency is decreased by congestion, which frequently results in service interruptions. In order to preserve the continuity of data flow in the network, it is necessary to design approaches and processes to avoid congestion or to minimize its effects. Congestion are avoided at two levels, knot and link, by using different techniques, including close loop or open loop. This paper proposed a new technique (TLBCA) to distribute packets arriving at a specific node on the links connected to it, which leads to the same destination, to avoid congestion on one of the links and out of service, which then leads to more load on the other links until the network collapses. TLBCA is based on the popular algorithm Round Robin. ECMP and TinyFlow, the two most widely used comparable algorithms in this sector, performed around 20% worse than the approach when it was simulated using OMNET++.

MSC..

https://doi.org/10.29304/jqcm.2022.14.4.1083

∗Corresponding author

Email addresses:

Communicated by 'sub etitor'

1. Introduction

Congestion is a great significant issue in computer networks that is may affect negatively the networks, which leads to slowness networks and reduced QoS (Quality of Service), and even stop it in many cases. It occurs when a network node or link is carrying more data than it can be handled and the load on the network is greater than the capacity of the network[1], in another word it occurs when more data is sent to the network than the network can handle [2][3].

Congestion is defined as a state of network elements where the network is unable to achieve network performance objectives for existing connections as well as new connection requests. It occurs when the amount of packets passing through a network segment is higher than usual; as a result, network performance decreases [2][4].

Congestion control refers to the set of actions taken by the network to minimize the intensity, spread, and duration of congestion or avoid it before it happens, such as communication links, buffers, and network switches. In other words, it refers to the machines and techniques that can either prevent congestion before it happens (congestion avoidance) or remove congestion after it happened [5][6]. One of the main objectives of congestion control is to allocate resources to shared networks. Therefore, it works on the network by several techniques to achieve an acceptable level of performance at networks and maximum bandwidth utilization [7].

Congestion control for a traffic load balance and packet switching over a computer network is being a big challenge due to the traffic sensitivity. These challenges represent a great motivation for researchers over the past decades to develop a large number of protocols for congestion control and avoidance to secure traffic for moving packets from their source to their destination without data corruption or packet loss [8].

The main objective of this review is to identify and understand how congestion occurs, and review different techniques and mechanisms for congestion control including open loop and closed loop techniques, and proposed a new method (TLBCA) to avoid congestion by using the traffic load balancing technique.

2. Congestion Control and Congestion Occurring

Congestion occurs when a network node or link is carrying data more than it can handle [3], and the load on the network is greater than the capacity of the network, in another word, It occurs when more data is sent to the network that the network cannot handle, where at some point, a point is generated in the network known as the bottleneck [9]. This point constitutes the main defect of the network, where the amount of data before this point is very large and after it is almost non-existent because the node or link is unable to deal with this data, thus causes a great slowdown in the network, and even stops it often. The congestion can be explained in the Fig. 1 and in below steps [8]:

1. The sender sends more data to the network than the network can handle.
2. Packet loss occurs at the receiver and the data is not completely received by the receiver.
3. The receiver asks the sender to send the data again.
4. A new load occurs on the network.
5. Thus, consecutive loads lead to network breakdown.

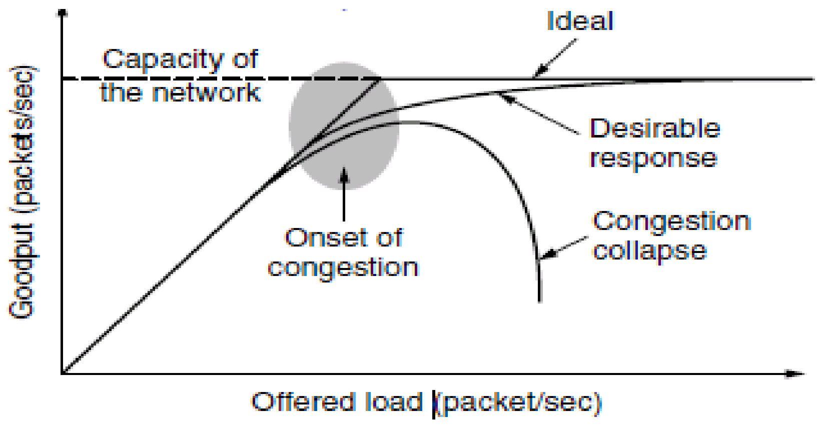


Fig. 1 - Congestion occurring

3. Type of Congestion control mechanisms and techniques

There are mechanisms followed to control congestion by either avoiding it before it occurs or (removes) it after it occurs they are open loop and close loop that are summarized in the diagram Fig. 2 as below [5][10]:

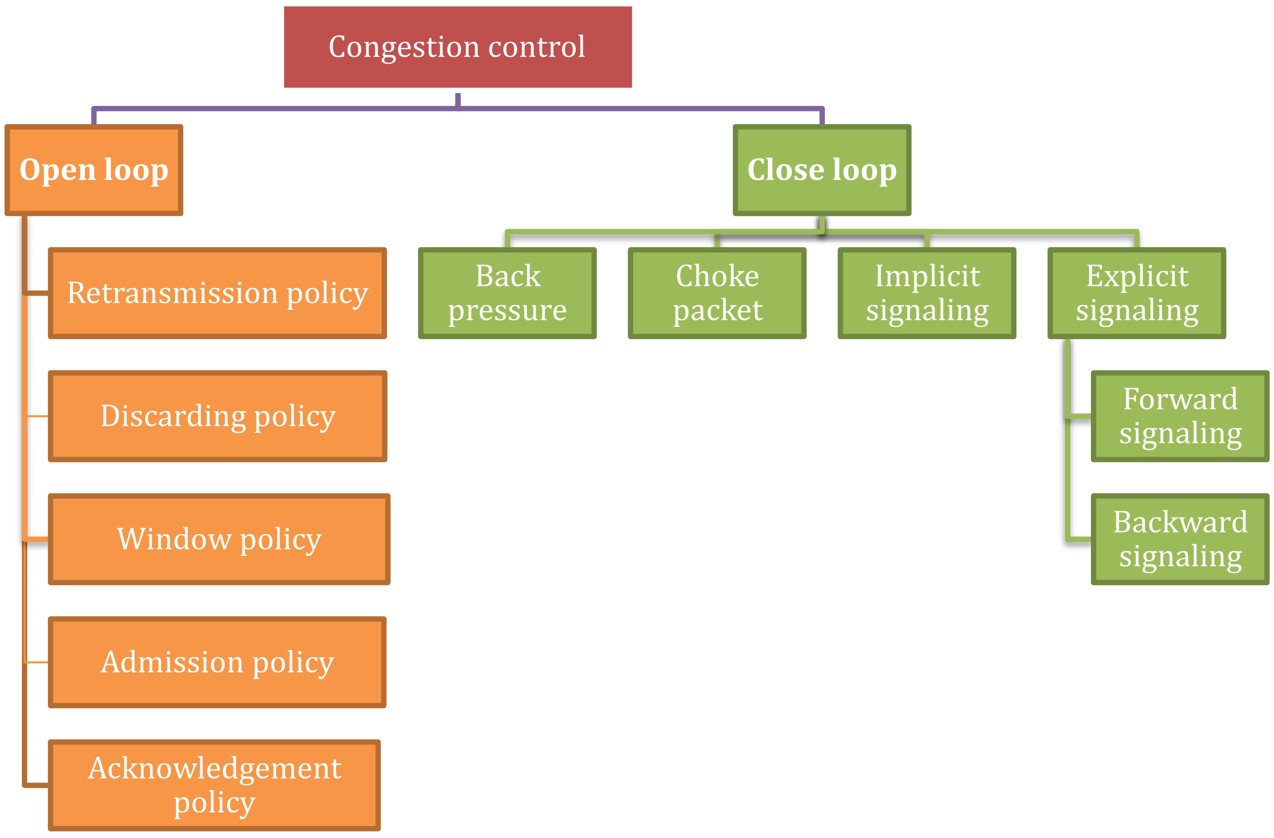


Fig. 2 - Congestion control mechanisms and techniques

3.1. Open loop congestion control techniques

It is a set of policies designed to avoid congestion before it occurs, it can be referred as congestion avoidance or Prevention of congestion and it can be define as one of the most important aspects of congestion control. It includes a computer network that is meant to lessen the chances of congestion. By applying the algorithms of prevention and it ought to incorporate preventing congestion routing/balancing algorithms and congestion control policies so that the user access rate does not surpass its participation in the rate of traffic and the assurance of crucial classes of traffic [9], [11]. It is includes:

3.1.1. Retransmission policy

Retransmission congestion control policy is working to handle the retransmission of the packets in the networks. If the sender received confirmation from the receiver that a packet that was sent has been lost or corrupted. Of course, the sender needs to re-send the data again [12]. The network may become more congested because of the actions. So to prevent congestion from happen this policy take an action to rest the time of retransmit the packet when the network have available resource.

3.1.2. Discarding policy

A discarding policy is used by a node in the network when the node discards the less sensitive packages to maintain the quality of service of the network. For example, the real-time audio and video have higher priority than the text files, so the node must discard or let text in query and pass the real-time files. all that to avoid the congestion from happening [13]–[15].

3.1.3. Window policy

Window policy is working when the receiver has received some packets successfully but not all of them, so the data is need to be transmitted again. This duplication of the data has the potential to cause network congestion; the window policy works to send only the missing data to reduce the amount of transmitted packets [16].

3.1.4. Admission policy

In admission policy, the node is checks the resource that required that is needs to transmit it data to the next node. If that node does not have enough resource to handle the data, this policy takes action to select another node or make a load balancing between more than one nodes that have connection with the destination of the data, to prevent make a congestion point in the network which is known as the bottleneck , fully use network sources and maintain data flow in the network [17], [18].

3.1.5. Acknowledgment(ACK) policy

The acknowledgment policy is used when the sender is send packet to the receiver, the sender should have should have heard from the receiver if the packet was received normally or not. These ACKs is part of traffic in the network. If the ACKs is too many it has may cause congestion in the network [19]. Several techniques are used to avoid congestion related to ACKs:

1. Instead of sending ACK for a one packet, the receiver can send ACKs for N packets.
2. Send an ACK only if the receiver needs to send a packet again or if the timer expires.

3.2. Close loop congestion control techniques

Closed loop congestion control techniques are used to treat or alleviate congestion after it happens. The efforts taken by the network after the discovery of performance corruption are known as congestion recovery. When congestion occurs, the network will be recovered due to the activities that have related to its creation. The goal of congestion recovery is to reduce congestion's effects while also restoring the network's normal operating status following congestion detection. When congestion is identified, networks without congestion recovery techniques may be totally damaged. Therefore, there is still a need for recovery plans, regardless of the possibility that network congestion adopts a strategy to prevent congestion. The clarification behind this is to keep effectiveness in the occasion of sudden modifications inside the network, which may bring about congestion[11],[20], [21]. It`s include:

3.2.1. Backpressure

Backpressure prevents a congested node from taking packets from the upstream node. As a result, the previous node or nodes may be congested and refuse to accept data from the node above. Only virtual circuits (VC's) with information about their upstream nodes can apply this congestion control mechanism[22], which spreads in the reverse direction of data flow from node to node. In Fig. 3, the third node becomes congested and stops accepting packets, causing the second node to become congested as the output data flow slows. In the same way, the first node may become overburdened and notify the source to slow down[23].

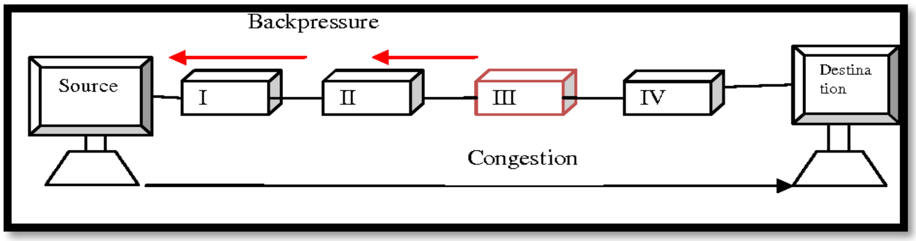


Fig. 3 - Close loop backpressure technique

3.2.2. Choke packet technique

A choke packet is a message delivered by a node to the source informing it that it is experiencing congestion. Each node keeps track of its resources and how many of its output lines are in use. When the resource use exceeds the administrator-defined threshold, the node sends a direct packet from the congestion node to the source, to give it feedback on how to minimize traffic. Fig. 4 shows that the intermediary nodes through which the packets have gone are congested and reported directly to the source to take action and slow down the transmission rate[24].

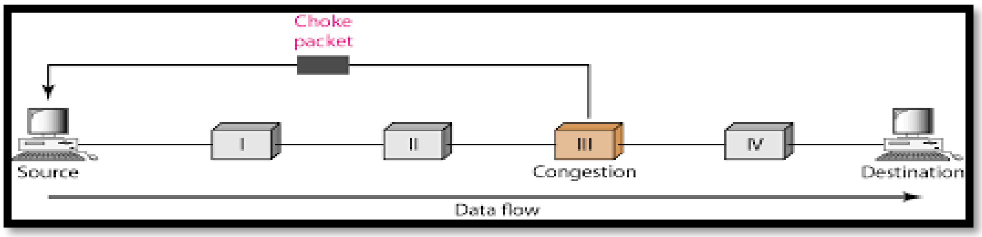


Fig. 4 - Close loop choke packet technique

3.2.3. Implicit signaling

In implicit signaling, the congestion node and its source do not have any direct communication with each other. The sender estimate that there is congestion in a network. For an instance, one assumption is that there is congestion in the network and the sender needs to reduce the amount of sending packets when a sender sends multiple packets and does not receive an ACK for a long time. [25].

3.2.4. Explicit Signaling

In explicit signaling, if the node gets congestion it can send direct a packet to the source or destination of the data to inform them about congestion. The difference between a choke packet and explicit signaling is that with explicit signaling, the signal is embedded in the data packets rather than being created as a separate packet as with choke packet [25][26].

Explicit signaling can occurs two direction:

1. Forward Signaling: It is a signal that transmitted in the direction of the congestion. There is a congestion warning for the destination. In this instance, the receiver sets policies to avoid the future congestion.
2. Backward Signaling: Sent signal in the reverse direction of the congestion. Due to road congestion, the source has been urged to slow down.

4. Proposed method

In this peper presents proposed approaches using traffic load balancing to avoid congestion before it has happened in end-to-end computer networks, which improves network performance and increases access to network resources in a complete and balanced manner.

Initially, the algorithms define all available paths to get from the source to the destination in a similar way to the work of the protocol Transmission Control Protocol/Internet Protocol (TCP/IP). In order to take into consideration all the available paths for the algorithm work, according to the data passing through the network and the capacity of those paths. For that purpose, they are needed local information such as the (delay time and bandwidth) of links, or (rate of packet dropping, buffer wait time and throughput) of nodes.

The proposed algorithm will be referred to as Traffic Load Balancing based on Congestion Avoidnice (TLBCA). The TLBCA is based on one of the most commonly used load balancing algorithms, which is Round Robin(RR), which balances the loads equally between two or more parties. For example, suppose there are requests (r) from a group of computers that request a service from the network, and there is more than one server to provide this service, and that all of these servers are equal in capacity. The algorithm works to transfer requests to the servers sequentially and equally, as shown in the Fig. 5 explains how the method TLBCA works in one device(node).

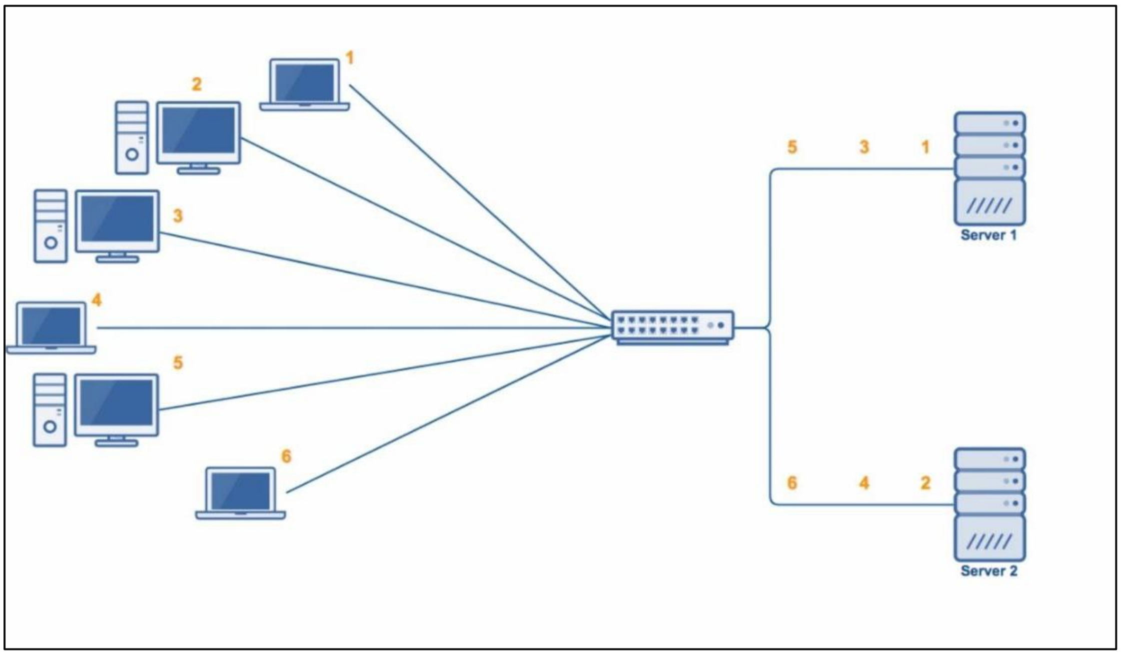


Fig. 5 - Illustration of the proposed method TLBCA working in one device

In the same way that requests are distributed to servers in the previous example, traffic are distributed to nodes according to the accessibility of the destination, where all links have the same capacity. In the proposed algorithm, the processing is carried out extensively to include the entire computer network, where the entire path from source to destination is studied end-to-end in order to reduce the burden generated on one node without the other and prevent the network from reaching the state of congestion and the occurrence of a bottleneck.

This algorithm works inside each node of the network nodes and distributes the traffic on the connection links that connected to the other nodes depending on the equal capacity of those links.

When the traffic reaches a certain node, it tests the possibility of this traffic reaching the destination through the links connected to those nodes. If there is only one link, the traffic goes directly to that link because there is no possibility of load balancing because it is one link. In the event when the traffic arrives at a node that has more than one connection in the network, and all of connection links lead to the destination, the algorithm directs the traffic packets towards the idlest link of least use, if the link accommodates all the traffic passing through it, then all the traffic is sent to the link. But if the link is not able to accommodates all the traffic, it means that the amount of data passing through is greater than the link's capacity, so the algorithm directs the remaining traffic to another link and also according to the idlest available link, and this process continues until the packets reaches the destination.

The algorithm works on the traffic coming to the network, where the traffic are directed to the destination through the best available path in terms of equally capacity and load. The algorithm works in a simple way and without many complications in assigning traffic packets to the available links that lead to the same destination in order to avoid congestion or a large difference in the consumption of resources of one path without the other, thus balancing the network load equally. It is worth noting that the algorithm works according to the common policy first-come-first-serve (FCFS) Without taking into consideration the type of packets passing through the network, with their different types, such as text, voice and image etc.., direct packets to the paths in succession according to the single queuing system.

Fig. 6 is flowchart shows how the algorithm works in the network.

Path detection: The sender knows the path to which the packet should be directed by knowing the possibility of this packet reaching its destination using the paths available in the node at the sender, and then chooses the path that is least used to distribute data equally between the paths to make a way to avoid the occurrence of crowding or an increase in the volume of data for a particular path.

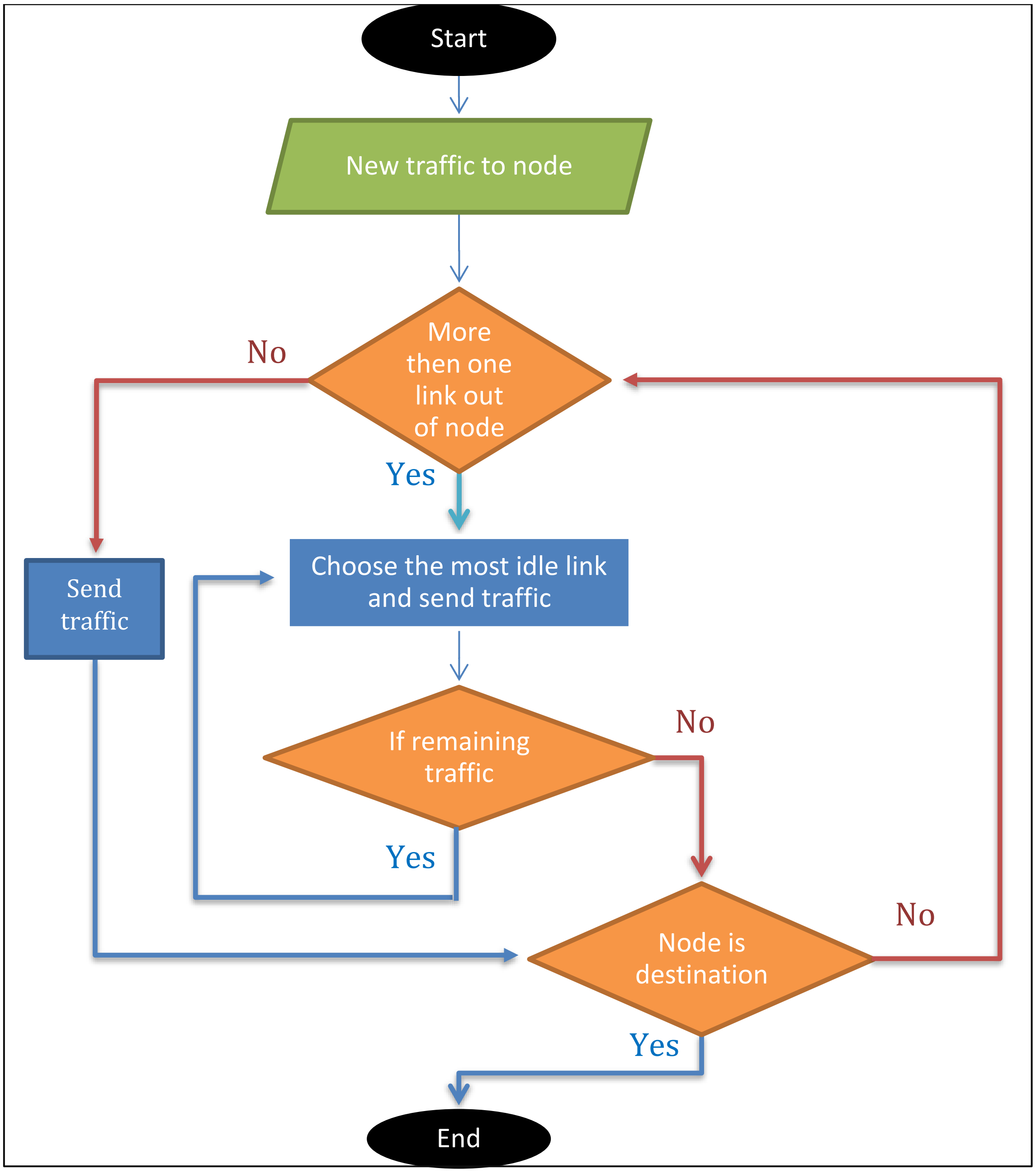


Fig. 6: Flowchart for the proposed method TLBCA

5. Experiment Results and Network Simulation

In this section, a network simulator used in to simulate the computer networks that have been used proposed method in order to apply it. The simulator program designed by Objective Modular Network Testbed in C++ (OMNeT++) version (5.6.2). The OMNeT++ simulator is use a graphical user interface that build on C++ program language. The proposed methods using simulate the computer networks (CN1, CN2) as shown in Fig. (7) and (9). Fig. (8) and (10) describes how to distribute packets for optimum use of network resources

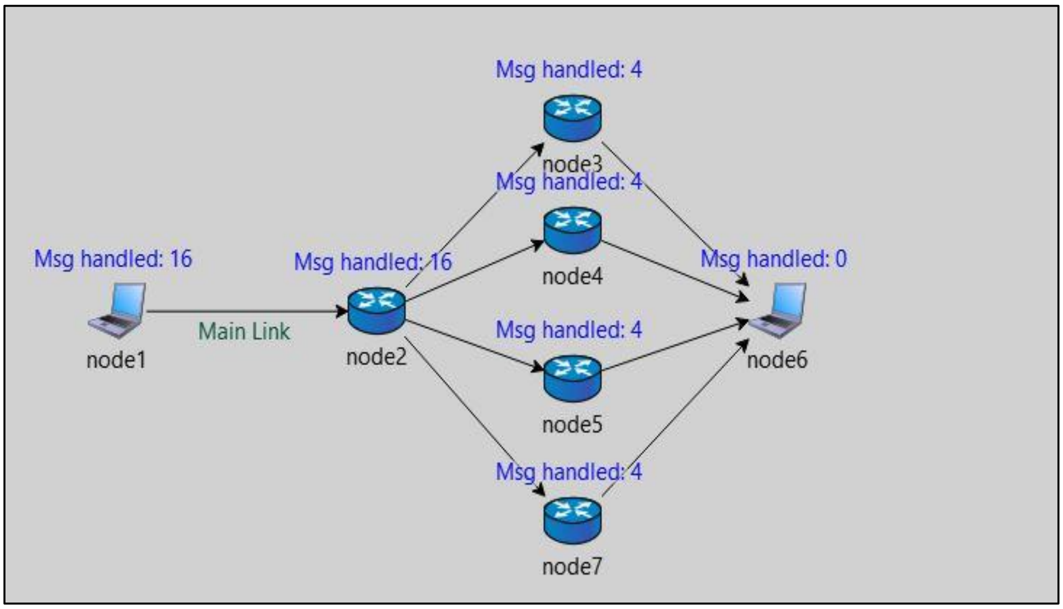


Fig .7 - Computer networks (CN1) simulated in OMNeT++

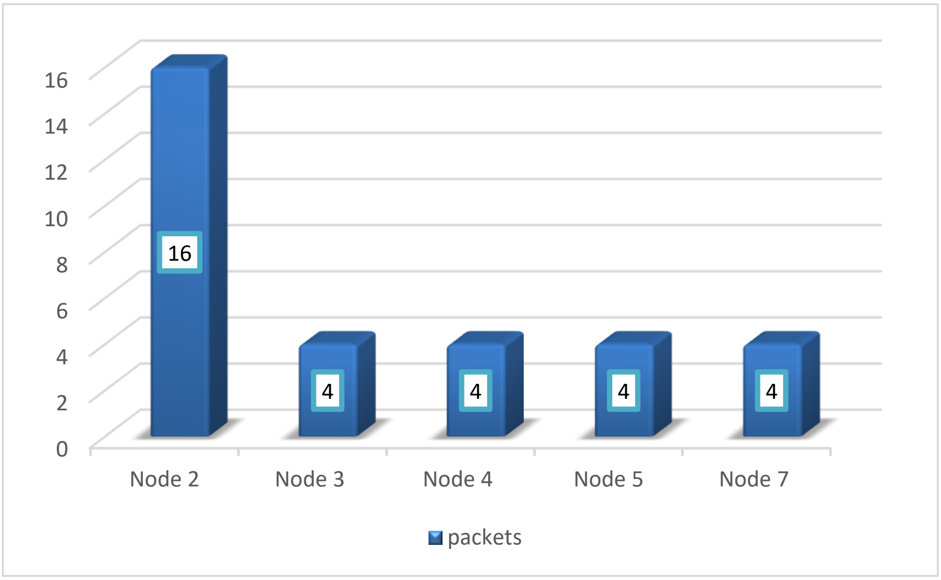


Fig. 8 Distribute of packets to nodes for Computer networks (CN1)

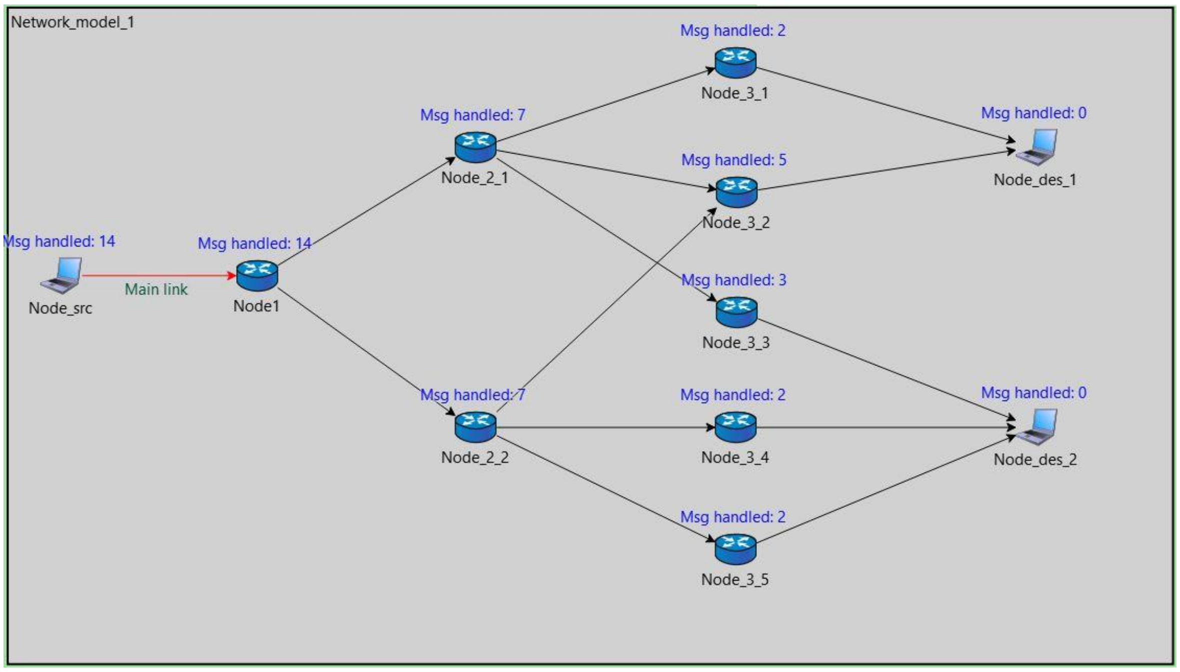


Fig. 9 - Computer networks (CN2) simulated in OMNeT++

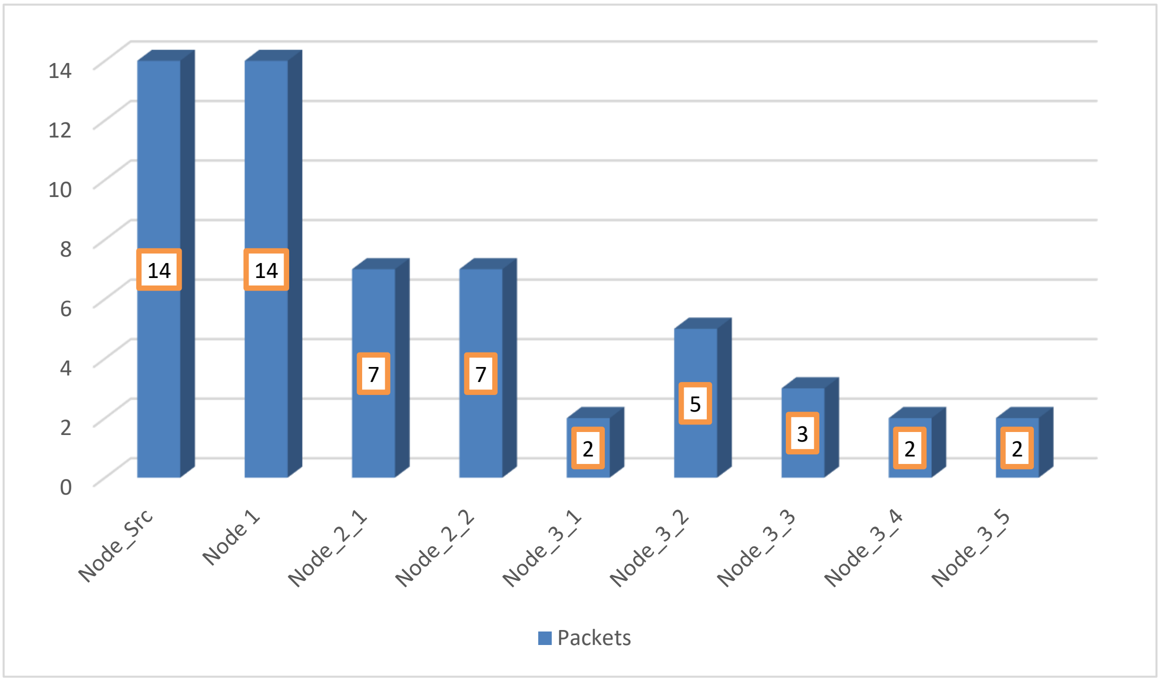


Fig. 10 - Distribute of packets to nodes for Computer networks (CN2)

We notice from figure 10 that the nodes that have more links consume more of their resources, as is the case in the node_3_2 that handles five packets. It is the highest among its peers of the same level.

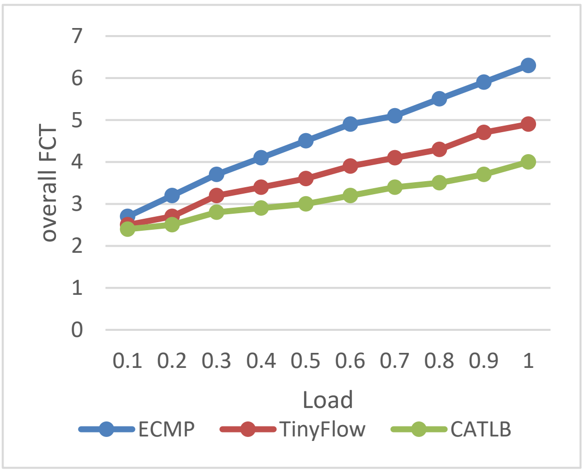
6. Performance Comparison

This section reviews a performance comparison between our proposed method and other method that based on traffic load balance to solve the congetion issue in particular it is based on the Round Robin algorithm, or a development of it. We will compare our work with two algorithms TinyFlow and Equal Cost Multipathing (ECMP) [27].

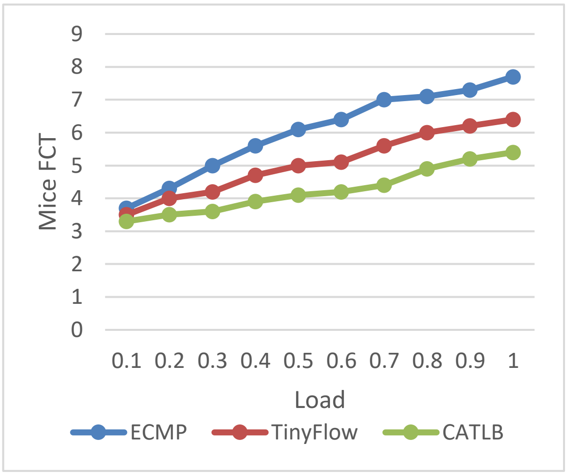
Overall Performance of the flow: flow completion times (FCT) performance. From Fig. 11 (a), TLBCA minmuze the mean FCT for all flows by up to 20% vs to TinyFlow. The load is starting low with 0.1 and 0.2, the improvement is small and simple as the path is uncongested at the first but the benefits of using TLBCA are larger when the load is going to grow and the possibility of having congestion is lager.

The Performance of Mice Flows: are the flow less than 100 KB as in the Fig. 11 (b) show the mean of micro flow. We can note from it that the improvement has become clearer in the proposed algorithm than its predecessors have where the improvement rate is 20%-25%.

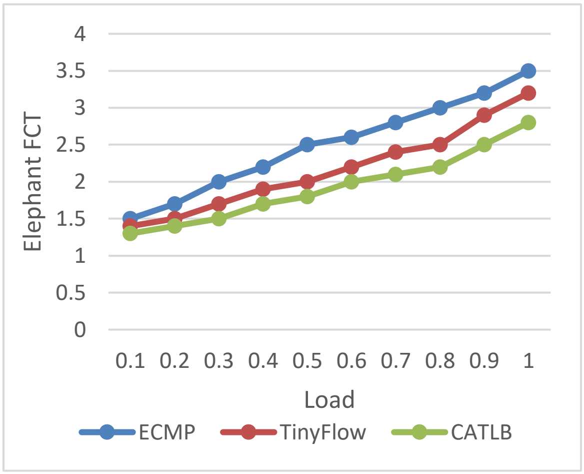
Performance of Elephant Flows: are the flow grater than 100 KB as in the Fig. 11 (c) show the mean of Elephant flow. TLBCA still improves the mean FCT by around 20%.



(a) All flows



(b) Mice Flows (0, 100] KB



(c) Elephant flows (100, 1) KB.

Fig. 11 - Mean FCT comparison between TLBCA, TinyFlow and ECMP

7. Conclusion

Congestion control is a serious field of study in computer networks and communications. Congestion control systems are difficult to build with minimal resources. This paper provides a comprehensive overview of current congestion management methods. The goal of all of the strategies is to extend the network's lifetime by utilizing limited resources. This research paper reviewed congestion and how to get rid of it in various ways and mechanisms and then suggested a specific method to avoid congestion by balancing the network load (TLBCA). TLBCA obtained good results with an average of 20% of the previous algorithms.

REFERENCES

[1] E. N. . Obinna and L. G. ** , Kabari, “Comparative Analysis of Drop Tail, Red and NLRED Congestion Control Algorithms Using NS2 Simulator,” *Int. J. Sci. Res. Publ.*, vol. 8, no. 8, 2018.

[2] J. F. Kurose and K. W. Ross, *Computer Networking A Top Down Approach*, Eighth edi. 2020.

[3] J. Chavula, “Improving Pan-African research and education networks through traffic engineering: A LISP/SDN approach,” 2017.

[4] K. A. Nasar, T. Y. Abdalla, and A. Y. Abdalla, “Computer Network Routing Using Fuzzy Systems,” *basrah J. Sci.*, vol. 26, no. 1A english, 2008.

[5] M. Welzl, *Network congestion control: managing internet traffic*. John Wiley & Sons, 2005.

[6] A. Y. Abdalla, T. Y. Abdalla, and K. A. Nasar, “Computer Network Routing Using Neural Networks,” *Basrah J. Sci.*, vol. 30, no. 1, pp. 1–14, 2012.

[7] X. Wang, *Scheduling and Congestion Control for Wireless Internet*. Springer, 2014.

[8] C. N. Houmkozlis and G. A. Rovithakis, *End-to-end Adaptive Congestion Control in TCP/IP Networks*. CRC Press, 2017.

[9] K. A. Nassar and A. A. Abdullah, “End-to-End Fuzzy RED to Reduce Packet Loss in Virtual Circuit Network,” *J. Univ. Babylon*, vol. 25, no. 3, 2017.

[10] S. Gupta, N. K. Sharma, and K. P. Yadav, “A Survey on Congestion Control & Avoidance.” VSRD-IJCSIT, 2012.

[11] A. S. Tanenbaum and D. Wetherall, “The network layer,” *Comput. Networks*, pp. 343–423, 2011.

[12] J. Liu, H. Zou, J. Dou, and Y. Gao, “Rethinking retransmission policy in concurrent multipath transfer,” in *2008 International Conference on Intelligent Information Hiding and Multimedia Signal Processing*, 2008, pp. 1005–1008.

[13] A. Pehlivanlı, “Software implementations of QoS scheduling algorithms for high speed networks.” Middle East Technical University, 2015.

[14] R. Kauser, “QoS aware packet scheduling in the downlink of LTE-Advanced networks.” Queen Mary, University of London, 2013.

[15] M. H. E. Mohamed, “Some active queue management methods for controlling packet queueing delay,” *Des. Perform. Eval. Some New Versions Act. Queue Manag. Schemes Control. Pack. Queueing Delay a Buffer to Satisf. Qual. Serv. Requir. Realt. Multimed. Appl.*, 2010.

[16] V. Jacobson, “Congestion avoidance and control,” *ACM SIGCOMM Comput. Commun. Rev.*, vol. 18, no. 4, pp. 314–329, 1988.

[17] S. J. Golestani, “A stop-and-go queueing framework for congestion management,” in *Proceedings of the ACM symposium on Communications architectures & protocols*, 1990, pp. 8–18.

[18] S. J. Golestani, “A framing strategy for congestion management,” *IEEE J. Sel. Areas Commun.*, vol. 9, no. 7, pp. 1064–1077, 1991.

[19] J. Waldby, U. Madhow, and T. V Lakshman, “Total acknowledgements (extended abstract) a robust feedback mechanism for end-to-end congestion control,” in *Proceedings of the 1998 ACM SIGMETRICS joint international conference on Measurement and modeling of computer systems*, 1998, pp. 274–275.

[20] D.-I. Choi and S.-M. Lee, “Performance analysis of the leaky bucket scheme with queue length dependent arrival rates,” *Bull. Korean Math. Soc.*, vol. 43, no. 3, pp. 657–669, 2006.

[21] G. Appenzeller, I. Keslassy, and N. McKeown, “Sizing router buffers,” *ACM SIGCOMM Comput. Commun. Rev.*, vol. 34, no. 4, pp. 281–292, 2004.

[22] S. Bolla, “Implementation of virtual circuits as a switching fabric in virtual modularized network.” University of Toledo, 2010.

[23] J. N. Odii, F. O. Nwokoma, T. U. Onwuma, and J. I. Eke, “NETWORK CONGESTION CONTROL SYSTEM USING FRAME RELAY TECHNOLOGY,” 2017.

[24] J.-N. Hwang, *Multimedia networking: From theory to practice*. Cambridge university press, 2009.

[25] K. M. A. Patel and R. Martolia, “Congestion control techniques in networking,” in *2016 International Conference on Communication and Signal Processing (ICCSP)*, 2016, pp. 1831–1835.

[26] A. Almeida and C. Belo, “Explicit congestion control based on 1-bit probabilistic marking,” *Comput. Commun.*, vol. 33, pp. S30–S40, 2010.

[27] J. Zhang, F. R. Yu, S. Wang, T. Huang, Z. Liu, and Y. Liu, “Load balancing in data center networks: A survey,” *IEEE Commun. Surv. Tutorials*, vol. 20, no. 3, pp. 2324–2352, 2018.