ISSN 2460-9056
http://socj.telkomuniversity.ac.id/ijoict/
Intl. Journal on ICT
Vol. 9, No. 1, Jun 2023. pp. 86-94
doi: doi.org/10.21108/ijoict.v9i1.716

Performance Comparison of TCP AFW and TCP FeW in IEEE 802.11 Wireless Networks: A Simulation Study

Arnas Sofyan[1], Vera Suryani[2], Hilal H. Nuha[3*]

[1] *Vodafone, Doha, Qatar*

[2,3*] *Faculty of Informatics, Telkom University, Bandung*
[*]hilalnuha@ieee.org

Abstract

Transmission Control Protocol (TCP), a layer 4 transport protocol, plays a crucial role in both wireless and wired networks. However, its performance in wireless networks is often unsatisfactory due to issues such as bandwidth limitations and utility problems with lower network layers. The mobility effect further exacerbates TCP's performance, as it fails to distinguish between connection failure and congestion-induced connection loss. In response to this challenge, researchers have explored potential solutions and found that TCP FeW outperforms the existing TCP NewReno. Building upon this background, this paper aims to simulate and analyze the performance of TCP AFW and TCP FeW in an IEEE 802.11 network. The simulations conducted using ns2 in a limited environment with random mobile scenarios reveal that TCP AFW achieves a 1.12% higher throughput compared to FeW, even with minimal modifications.

Keywords: Transmission Control Protocol, FeW, AFW, ns2

I. INTRODUCTION

Wireless LAN technology has witnessed significant growth and is widely used in various network applications such as ad hoc networks, sensor networks, wireless mesh networks, and home/office networks. However, wireless LANs differ from wired networks in several aspects, including shared queues, half duplex links, channel noise, and mobility effects. While TCP plays a crucial role in both wireless and wired networks, its performance in wireless networks is often unsatisfactory, leading to issues such as bandwidth limitations and utility problems with lower network layers [6]. Research indicates that the mobility effect further degrades TCP performance in wireless networks. This is primarily because TCP lacks the ability to distinguish between connection failure and congestion, resulting in connection loss, commonly referred to as wireless link loss. In TCP, each connection between a client and server involves a single stream.

However, this approach poses a challenge as data transmission blocks suffer losses at each point, which is acceptable when transmitting text but not suitable for real-time data like audio or video. Certain ad hoc network applications, which rely on wireless LAN-based hotspots, require the support of TCP in a multihop topology to serve clients across a wireless network. Previous studies have demonstrated the superiority of TCP FeW over TCP Reno in terms of throughput parameters and data transmission efficiency in a multihop Mobile Ad Hoc Network environment on a wireless network. Notably, this optimization is achieved without altering the fundamental TCP mechanism but by emphasizing a smoother and slower increase in the window growth rate [2].

Received on 31 May 2023. Revised on 23 June 2023. Accepted and Published on 25 July 2023.

In TCP AFW set the value of increasing the number of cwnd which is more conditional than FeW. What is meant by conditional here is that with this AFW method, TCP can adapt to network conditions when it is crowded or not [1]. Based on this, the performance of TCP AFW and TCP FeW will be compared in terms of throughput and packet loss parameters. Which one has better performance to improve TCP performance on end-to-end service on IEEE 802.11 network.

II. LITERATURE REVIEW

A. Existing Research

In their work [2], the authors conducted an analysis of TCP in a multihop 802.11 network, focusing on inter-layer research. The study aimed to examine the impact of congestion and MAC contention on the interaction between TCP and dynamic source routing protocols in an 802.11 ad hoc network. The findings revealed that issues arose due to a lack of coordination and sharing within the network. Based on the observations made during the study, it was concluded that the primary source of TCP problems in 802.11 ad hoc wireless networks stemmed from the Windows TCP mechanism itself. To address this issue, the authors proposed a solution called the Fractional Windows Increment (FeW) scheme, which aimed to mitigate the aggressive nature of TCP. The proposed scheme was defined by an equation, which specified the modifications to be made to the existing TCP mechanism.

$$W^{new} = W^{Current} + \frac{\alpha}{W^{current}} \quad (1)$$

With conditions where $0<\alpha<1$, with a value of < 1, it is expected to limit the aggressive nature of TCP, and it is proven to be successful in increasing delivery efficiency and throughput. However, there is a drawback in this mechanism, namely by setting a small window growth rate, discarding or not utilizing bandwidth before the congestion window increases with BDP, this causes fast timeouts when the window size is equal to or greater than BDP. Based on the results of the previous research, [1] proposed a new scheme to improve the performance of FeW itself, namely the Adaptive Fractional Window Increment (AFW) scheme. AFW utilizes bandwidth when the network is idle, and limits windows operating when the network is congested. In this case AFW not only limits the aggressive nature of TCP but also makes it more adaptive to its environment in this case the network state. In TCP AFW set the value of increasing the value of cwnd which is more aggressive than FeW. The cwnd increase in AFW is updated when a timeout occurs, which indicates the network is congested and the cwnd size is larger than the bandwidth delay product (BDP) [1]

B. Theoretical Foundation

1) Transport Control Protocol

The transport control protocol (TCP) [13] was first developed in 1974 by Bob Kahn and Vinton Cerf (ACM Pressroom, 2005). They started the project in the 1970s When ARPAnet, a project developed in the United States related to research on military networks, was developed. The beta version of ARPAnet used the Network Control Protocol (NCP) to operate a network, but in 1973.

2) Transport Control Protocol Layer

Protocols in ordinary networks are formed in layers. Each layer has its own function (Stevens, 1993). Stevens describes the arrangement of the TCP/IP protocol layers.

3) TCP Service

The network layer is used by TCP and UDP [14]. However, at the transport layer, both have different services. TCP provides connection-oriented and reliable services. Connection oriented means that when two applications are connected, the two applications must establish a connection before they exchange data [12].

4) TCP Flow Control

It was previously discussed that TCP provides reliable service between 2 communicating hosts. TCP also uses flow and congestion control. This is related to the use of a link between 2 hosts, where TCP always optimizes data transmission until it reaches its optimal capability, but it will reduce the packets sent if the receiving host cannot handle it [8].

5) TCP Congestion Control

The TCP congestion control algorithm has 3 main components, namely: slow start, congestion avoidance, and fast recovery. Slow start and congestion avoidance are the main elements of TCP, both of which regulate the size of the congestion window (cwnd) in response to received acknowledgments. *Fast recovery* itself is a useful additional step to fix the size of cwnd, which is a variable that describes the size of congestion windows.

6) Optimization of TCP FeW with AFW Algorithm

Optimization of FeW is then carried out by applying the AFW algorithm in it, this is due to the lack of FeW which sets the value or rate of increasing the amount of cwnd that remains at the time of a timeout as described in the previous section. If the implementation in FeW, the Increment mechanism when a timeout occurs is:

$$Increment = \frac{\alpha}{w^{current}} \tag{2}$$

7) Test Parameters

- Dropped Packet

"Dropped Packets" refers to the quantity of packets discarded by the router when facing an overload of incoming packets or to signify congestion, commonly implemented in Active Queue Management (AQM) techniques. This crucial performance metric is also known as the "packet loss rate" and can be mathematically represented by the following equation. Monitoring and understanding the packet loss rate are essential in assessing network efficiency, pinpointing potential congestion concerns, and ensuring smooth data transmission within the system.

$$Loss\ Rate = \frac{Number\ of\ Dropped\ Packet}{Number\ of\ Submitted\ Packet} \times 100\% \tag{3}$$

- Throughput

Throughput, a critical metric in networking, quantifies the overall volume of bytes received by the receiver during the time span encompassing the transmission of the first and last packets. This measure can be represented by equation (3), signifying its importance in assessing the efficiency and data handling capabilities of the communication system.

$$Throughput = \frac{Number\ of\ Received\ Packet}{Observation\ Duration} \times 100\% \tag{3}$$

C. TCP Version

The first version of TCP was described in RFC 793 in 1981. The document describes the basic elements of the protocol, such as the sliding windows algorithm, header format and retransmission timer, i.e. sending back packets after no acknowledgment has been received from a packet at a time. TCP Reno is the most used at this time. Paper [8] concluded that there are interesting things about congestion control which contains 3 components, which include: slow start, congestion avoidance, and fast recovery. Suppose the slow start period is ignored. When the connection occurs for the first time, then it is assumed that the lost

packets are caused by 3 twin acknowledgments not from timeouts, then TCP congestion control increases the congestion window linearly by 1 MSS (maximum segment size) per RTT (additive increase) and divides by 2 the value of the congestion window when there are 3 twin acknowledgments (multiplicative decrease). Therefore, on top of TCP congestion control, it is often called the Additive-Increase, Multiplicative-Decrease algorithm.

III. RESEARCH METHOD

A. *Installation and Simulation of TCP AFW and FeW*

Upon completing the network configuration, the implementation of TCP AFW and FeW methods will commence. As guided by previous research [2], a modest value of 0.01 will be applied, followed by the incorporation of Smax and Smin parameters, alongside their corresponding values, into TCP AFW. This approach aims to optimize the performance of the TCP AFW method and facilitate a comprehensive comparison with the TCP FeW method. By carefully fine-tuning these parameters, the study seeks to gauge the effectiveness and efficiency of both approaches in the specified network setting in Table I.

TABLE I
LINK MODELING

No	Items	Spesification
1	IEEE 802.11	B
2	Media	Radio Wave
3	Rate	2 Mbps
4	Propagation radio	two-ray ground
5	Transmission Range	250 x 550 meters
6	Carrier Sensing Range	550 meters

B. *System Modeling*

Following the completion of the configuration and settings, the next step involves defining the measurements and replications that will be utilized for the simulation, as detailed in Table II. These measurements and replications are carefully selected to accurately assess the performance and behavior of the TCP AFW and TCP FeW methods under various network scenarios. By incorporating appropriate metrics and conducting multiple replications, the study aims to ensure the reliability and validity of the simulation results, enabling a robust evaluation of the two methods in the given network environment.

TABLE II
MODELING NODES

No	Items	Spesification
1	Mobile Host	Wireless
2	Service	FTP
3	Number of Hosts	50
4	Topology	Mobile Random
5	FTP Service Size	100 MB
6	TCP Packet Size	1024 bytes
7	Routing Protocol	DSR

C. *Running the Simulation*

After the topology, configuration, method implementation, and replication have been determined, the simulation is ready to run. When the simulation is running, you can see the message transfer process between modules. In this TCP AFW simulation, the simulation process can be stopped when the number of bytes that must be sent has arrived entirely at the receiving host.

D. Simulation Results

After the simulation process has been stopped, a simulation result file will be generated, namely a trace file. These files are then aggregated by file type and by the TCP type used in the simulation. After being grouped, the values associated with this experiment were observed.

E. System Device Requirements

1) Hardware Requirements (Hardware)

To design this system, specific hardware components are required. The essential hardware includes a computer with the following specifications: an operating system of Ubuntu 10.04, a Linux 2.6 kernel, an Intel Core i3 processor, 3GB DDR2 RAM, a 320GB hard disk, and a Fast Ethernet Card with a speed of 100mbps. These hardware components form the foundation of the system, providing the necessary computing power and storage capacity to support the intended functionalities. With this configuration, the system can effectively handle the tasks and requirements it is designed for.

2) Software Requirements

The experiment involved the utilization of specific software components. The software used in the experiment comprised Linux Ubuntu 10.04 as the operating system and Network Simulator 2 (NS2) [15] as the network simulation tool. Linux Ubuntu 10.04 provided a stable and reliable platform for conducting the experiment, while NS2 served as a powerful tool for simulating and analyzing network behavior. By employing these software components, the experiment was able to simulate various network scenarios and evaluate the performance of the system under different conditions.

3) Network Topology

In this section, we elucidate the process of creating the random ad hoc mobile network topology model, as depicted in Fig. 1. We begin with the solution representation, which serves as the foundation for generating a solution. The model encompasses various elements and parameters that collectively contribute to shaping the network topology [16]. By carefully defining the representation of the network, we can effectively produce a solution that accurately reflects the dynamic nature of ad hoc mobile networks. This comprehensive approach ensures that the generated topology is representative of real-world scenarios, facilitating a meaningful analysis of the network's performance and behavior in the subsequent stages of the study.

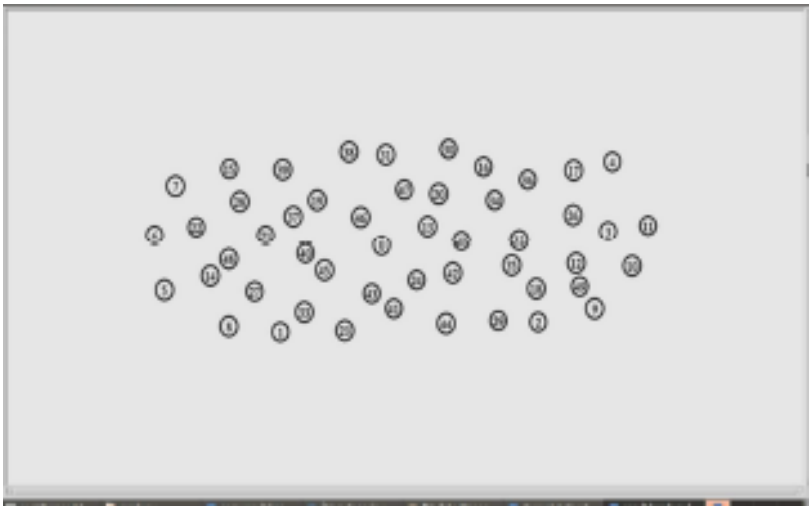

Fig. 1. Network Topology

F. Testing Scenarios

1) Purpose

With the aim of scrutinizing the performance of TCP AFW and TCP FeW in end-to-end services, we undertake the simulation of a network that incorporates both of these algorithms. By emulating this network environment, we can effectively evaluate and compare the effectiveness of TCP AFW and TCP FeW in handling end-to-end communication. This simulation enables us to gather valuable insights into the strengths and weaknesses of each method, thereby shedding light on their suitability for real-world applications and diverse network scenarios. The results obtained from this comparative analysis will contribute to advancing our understanding of these algorithms and their potential impact on enhancing end-to-end service delivery in networking systems.

2) Scenario

Based on the objectives of the scenario above, it is divided into 2 parts, namely:

a. Given AFW performance in terms of throughput and packet loss with a combination of $\alpha = \{2.4.6.8\}$ and $\beta = \{2.4.6,8\}$
b. Knowing the performance of FeW in terms of throughput and packet loss with a value of $\alpha = 0.01$

IV. RESULTS AND DISCUSSION

Some of the tests carried out were selected 12 combinations of and parameters from AFW and one FeW scenario that was already the most optimal based on previous research [2]. as a dataset the combination of and scenario is used.

A. Analysis of AFW. Parameter Observation Results

TABLE III
COMBINATION OF AND FOR OPTIMUM FEW AND AFW

Adaptive Fractional Window Increment	
$\alpha = 4$	$\beta = 8$
Fractional Window Increment	
$\alpha = 0.01$	

In this paper, there is a comparison of the largest throughput between AFW and FeW at flow N = 50 which is better at 1.12%. This value is obtained by comparing the average AFW throughput with FeW throughput. And the comparison of the value of the largest AFW and FeW throughput scenarios is found in the AFW value = 2 and = 2 at the total flow N = 50 of 1.16%. In previous research [1], the average throughput value comparison was 5% with different environments and different topologies.

TABLE IV
FEW AND AFW THROUGHPUT RESULTS

No	**Scenarios**	**Throughput (Kbps)**		
		и = 10	**и = 30**	**и = 50**
1	AFW $\alpha = 2\ \beta = 2$	1851.84	1935.18	2014.01
2	AFW $\alpha = 2\ \beta = 4$	1815.32	1932.42	1913.95
3	AFW $\alpha = 2\ \beta = 6$	1857.86	1898.32	1796.42
4	AFW $\alpha = 2\ \beta = 8$	1875.78	1898.32	1796.42
5	AFW $\alpha = 4\ \beta = 2$	1907.46	1874.39	1908.29
6	AFW $\alpha = 4\ \beta = 4$	1867.95	1999.84	1951.58
7	AFW $\alpha = 4\ \beta = 6$	1939.08	1958.84	2011.36
8	AFW $\alpha = 4\ \beta = 8$	1874.83	1958.84	1969.78
9	AFW $\alpha = 6\ \beta = 2$	1860.42	2020.90	1933.03
10	AFW $\alpha = 6\ \beta = 4$	1953.56	1977.11	1999.50
11	AFW $\alpha = 6\ \beta = 6$	1887.60	1903.17	2013.37
12	AFW $\alpha = 6\ \beta = 8$	1862.77	1913.12	2013.37
13	FEW $\alpha = 0.01$	1712.48	1878.02	1739.35

The largest comparison between FeW and AFW occurred in the simulation with the number of flows N = 10, which showed FeW was better at handling the Loss rate of 154%, this value was obtained by comparing the average Loss rate of AFW with the Loss rate of FeW. The comparison of the largest Loss rate scenario between AFW and FeW occurs in the AFW scenario with a value of = 6 and = 8 at N = 50 of 34.33%. When viewed from the mechanism that works on FeW by setting a low value of windows growth rate, namely by setting conditions $0<\alpha<1$ in idle or congested network conditions.

TABLE V
FEW AND AFW LOSS RATE RESULTS

No	Scenarios	Throughput (Kbps)		
		и = 10	**и = 30**	**и = 50**
1	AFW $\alpha = 2\ \beta = 2$	1.08	1.55	3.27
2	AFW $\alpha = 2\ \beta = 4$	0.97	1.68	3.92
3	AFW $\alpha = 2\ \beta = 6$	0.88	2.15	2.96
4	AFW $\alpha = 2\ \beta = 8$	1.05	2.15	3.11
5	AFW $\alpha = 4\ \beta = 2$	1.88	2.31	3.98
6	AFW $\alpha = 4\ \beta = 4$	1.54	2.73	3.92
7	AFW $\alpha = 4\ \beta = 6$	1.65	2.63	3.44
8	AFW $\alpha = 4\ \beta = 8$	1.76	2.63	3.56
9	AFW $\alpha = 6\ \beta = 2$	1.87	2.52	5.09
10	AFW $\alpha = 6\ \beta = 4$	2.22	3.16	4.11
11	AFW $\alpha = 6\ \beta = 6$	1.65	3.09	5.15
12	AFW $\alpha = 6\ \beta = 8$	1 97	3.28	5.15
13	FEW $\alpha = 0.01$	0.01	0.08	0.15

While the characteristics of AFW are more conditional, AFW utilizes bandwidth by multiplying the windows growth rate by when the network is idle, and limiting windows operating when the network is crowded by dividing the windows growth rate by by making maximum and minimum growth limits, with variables Smin and Smax as stated in Equation 2.

B. Based on the Optimum and Values on AFW

TABLE VI
COMBINATIONA AND FOR OPTIMUM AFW THROUGHPUT

AFW Scenarios	Throughput (Kbps)
	Flow и = 50
$\alpha = 2\ \beta = 2$	2014.01

During the simulation process, a noteworthy observation emerged. Specifically, when examining the Throughput metric with various parameter combinations (as displayed in Table VI), it became evident that certain combinations outperformed others, such as the combination (value X) compared to the combination (value Y). This finding contradicted the results reported in the literature reference [1], where the impact of and values on handling loss rates had not been measured. However, in this paper, we dedicated efforts to simulate and analyze the influence of and values on the loss rate, which are showcased in Table VII. The simulation results demonstrated that TCP AFW exhibited the most optimal handling of the loss rate when compared to other algorithms. This novel insight underscores the significance of exploring and understanding the effect of different parameter settings, ultimately enhancing the overall comprehension and applicability of TCP AFW in real-world network scenarios.

TABLE VII
COMBINATIONA AND FOR OPTIMUM AFW THROUGHPUT

AFW Scenarios	Loss Rate (%)
	Flow и = 10
$\alpha = 2\ \beta = 6$	0.88%

Analyzing Figure 2 reveals a notable trend: the throughput of the simulation consistently decreases over the course of the simulation duration. This decline can be attributed to the accumulation of data exchanges, leading to an increased volume of sent and retransmitted data. Consequently, this influx of data places a strain on the link capacity, causing a reduction in the network's ability to handle data flows efficiently. As the link capacity diminishes, the throughput achieved by the system experiences a corresponding decrease. This observation underscores the impact of data exchange duration on overall throughput, emphasizing the need for effective strategies to manage and optimize data transmission to maintain desirable performance levels in the network.

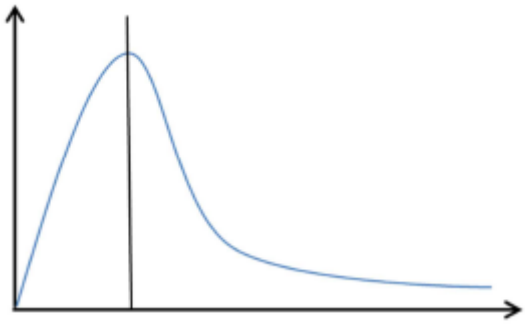

Fig. 2. Throughput with Respect to Simulation Time

V. CONCLUSION

After conducting the simulation, testing, and analysis, several key conclusions can be drawn. Firstly, TCP AFW demonstrates its implementation capabilities without necessitating changes to the fundamental TCP mechanism. Across all test scenarios, the AFW scheme exhibits superior performance in terms of the throughput parameter, achieving a 1.12% increase compared to other methods. On the other hand, the FeW mechanism, with its unique window size increase determined by the window growth rate ($0 < \alpha < 1$) in idle or congested network conditions, proves to be more adept at handling the loss rate, boasting the lowest loss rate of 0.1%. Although AFW experiences a higher number of dropped packets in simulation results, this outcome is reasonable given its more aggressive adaptation to network conditions. Despite this, AFW still transmits more packets (including retransmissions) than FeW [1]. Notably, AFW consistently produces better throughput values than FeW, aligning with previous research on Mobile Ad Hoc Networks. Upon completing this work, valuable suggestions are put forth to enhance future systems. Researchers are encouraged to explore simulations on different topologies within IEEE 802.11 networks, particularly in Mobile Ad Hoc Networks. Additionally, evaluating routing protocols on various wireless networks could yield valuable insights. Furthermore, the development of a new algorithm for TCP AFW is recommended to address its weaknesses in handling packet loss more effectively. Such advancements in algorithms can contribute to the continual improvement of TCP AFW, bolstering its performance and resilience in diverse network scenarios.

REFERENCES

[1] DING, Liang-hui, ZHANG, Wen-jun, WANG, Xin-bin, QIAN, Liang, XU, You-yun, 2008, Adaptive Fractional Window Increment of TCP in multihop ad hoc networks.

[2] Nahm, K, Helmy, A, and Jay Kuo, C, 2005, TCP over Multihop 802.11 Networks : Issues and Performance Enhancement, Proceeding of ACM MobiHoc '05, Urbana-Champaign, Illinois, USA, 277-287.

[3] Nahm, K, Helmy, A, and Jay Kuo, C, 2008, Cross-Layer Interactions of TCP Ad Hoc Routing Protocols in Multihop IEEE 802.11 Networks, IEEE Transaction on Mobile Computing, vol. 7, no. 4,

pp. 458-459.

[4] YU, X, 2004, Improving TCP Performance over mobile Ad Hoc Networks by exploiting Cross Layer Information Awareness, Proceedings of ACM MobiCom '04, Philadelphia, Pennsylvania, USA, pp 231-244.

[5] Kristiadi, Andriyas Danu Hari, Simulation of TCP Fractional Window Increment (FeW) in IEEE 802.11 Multihop Network.

[6] Holland, G., Vaidya, N., 1999. Analysis of TCP Performance over Mobile Ad Hoc Networks. proc. IEEE/ACM MobiCom, p-219-230.

[7] Johnson, D., Maltz, D., Hu, Y., 2004. The Dynamic Source Routing for Mobile Ad Hoc Networks IETF Internet Draft Available fromhttp://www.ietf.org/internet-drafts/draftietf-manet-dsr-10.txt

[8] Kurose, J., F., and Ross, KW, 2009, Computer Networking: A Top-Down Approach, 5th Edition, Addison Wesley Longman Publishing Co., Inc., Boston, Massachusetts, USA.

[9] Peterson, LL, and Davie, B, S., Computer Networks: A System Approach, 4th Edition, Morgan Kaufmanm, San Francisco, California, USA.

[10] RFC 2001, 1997, TCP Slow Start, Congestion Avoidance, Fast Retransmit, and Fast Recovery Algorithm, The Internet Engineering Task Force (IETF), Washington DC, USA. [11] RFC 793, 1981, Transmission Control Protocol, The Internet Engineering Task Force (IETF), Washington DC, USA.

[12] Stevens, WR, TCP/IP Illustrated, Volume 1: The Protocols, ed. 1, Addison Wesley Longman Publishing Co., Inc., Boston, Massachusetts, USA.

[13] Khan, T., Sohail, A., Qureshi, K. N., Iqbal, S., & Jeon, G. (2022). Multipath transport control protocol for 5G mobile augmented reality networks. International Journal of Communication Systems, 35(5), e4778.

[14] Mahmoodi Khaniabadi, S., Javadpour, A., Gheisari, M., Zhang, W., Liu, Y., & Sangaiah, A. K. (2023). An intelligent sustainable efficient transmission internet protocol to switch between User Datagram Protocol and Transmission Control Protocol in IoT computing. Expert Systems, 40(5), e13129.

[15] Ghaffarian, Hossein, and Mahdi Sadeghizadeh. "Parsim: A parametric simulation application for wireless sensor networks based on NS2 simulator." International Journal of Nonlinear Analysis and Applications (2022).

16] NUHA, Hilal, et al. Enhanced Multipath TCP to Improve the Mobile Device Network Resiliency. International Journal of Computing and Digital Systems, 2023, 14.1: 1-xx.