Cross-Layer based TCP Performance Enhancement in IoT Networks

Sultana Parween, Syed Zeeshan Hussain
Department. of Computer Science
Jamia Millia Islamia
New Delhi, India

***Abstract*—Transmission Control Protocol (TCP) used multiple paths for performing transmission of data simultaneously to improve its performance. However, previous TCP protocols in Internet of Things (IoT) networks experienced difficulty to transmit a greater number of subflows. To overcome the above issues, we introduced cross-layer framework to perform efficient packet scheduling and congestion control for increasing the performance of TCP in IoT networks. Initially, the proposed IoT network is constructed based on grid topology using Manhattan distance which improves the scalability and flexibility of the network. After network construction, packet scheduling is performed by considering numerous parameters such as bandwidth, delay, buffer rate, etc., using fitness based proportional fair (FPF) scheduling algorithm and selecting best subflow to reduce the transmission delay. The scheduled subflow is sent over an optimal path to improve the throughput and goodput. After packet scheduling, congestion control in TCP is performed using cooperative constraint approximation 3^+ (CoCoA3^+-TCP) algorithm in which three stages are employed namely congestion detection, fast retransmission, and recovery. The congestion detection in TCP-IoT environment is performed by considering several parameters in which cat and mouse-based optimization (CMO) is utilized to adaptively estimate retransmission timeout (RTO) for reducing the delay and improving the convergence during retransmission. Fast retransmission and recovery are performed to improve the network performance by adjusting the congestion window size thereby avoiding congestion. The simulation of cross-layer approach is carried out using network simulator (NS-3.26) and the simulation results show that the proposed work outperforms high TCP performance in terms of throughput, goodput, packet loss, and transmission delay, jitter, and congestion window size.**

***Keywords*—Internet of things (IoT); transmission control protocol (TCP); cross-layer approach; packet scheduling; congestion control; fast retransmission; recovery**

I. Introduction

In recent days, Internet of Things (IoT) is an emerging technology that provides connectivity between various heterogeneous devices to send data through the network. Various sensor nodes act as IoT devices that are interconnected and it is applied for various applications such as health care, home automation, environmental monitoring, transportation, and management of infrastructure [1], etc. Several IoT protocols are used for performing data transmission between heterogeneous IoT devices such as AMQP (advanced message queuing protocol), MQTT (message queuing telemetry transport) and CoAP (constrained application protocol), HTTP (HyperText Transfer Protocol), and XMPP (Extensible Messaging and Presence Protocol). These protocols are provided effective communication among IoT devices that are used in applications such as embedded systems and industrial 4.0 [2]. However, various challenges are present in IoT data transmission between sensor nodes due to their individual transmission rate which leads to high network congestion [3]. To overcome such challenges Transmission Control Protocol (TCP) which is a transport layer protocol is introduced for efficient data transmission because it transmits huge amount of data traffic globally and is constrained to handle network blockage by reducing the congestion in the IoT environment and also enhances the quality of service (QoS) [4], [5]. This protocol provides effective data transmission with better performance in terms of load, connectivity, reliability, low latency, and speed when compared with user datagram protocol (UDP) [6], [7].

However, several challenges are present in TCP protocol which affects the data transmission by fixed round trip time (RTT) and retransmission timeout (RTO), transmission delay, high buffer rate, bandwidth consumption, etc. that leads to high data congestion. The challenges in TCP affect the performance of the network [8], [9]. Several approaches are introduced to control the congestion occurred in TCP by implementing CoCoA, CoCoA$^+$, etc. algorithm [10] for classifying weak RTT. However, these algorithms are suffered from severe issues in affecting the network performance, so there would be an improvement in those algorithms must be needed and perform exponential backoff by estimating RTT and RTO however, the frequent retransmission in the network leads to congestion in the network [11]. Congestion control is performed by adjusting the size of congestion window (CWND) after successful recovery. The precise changes in congestion window with respect to the available bandwidth improve the throughput. Some of the works focused on adjusting the congestion window size by considering only limited assumptions. However, the precise analytical justification was not provided which affects the reliability [12]. Retransmission is performed if RTO is expired. After duplicate acknowledges, retransmission is performed. Several works need high retransmission attempts for better recovery [13]. Packet scheduling is performed to avoid congestion by selecting optimal paths and sub-flows [14]. Queue scheduling and path scheduling are also performed for congestion avoidance [15]. Several challenges are present in packet scheduling such as high consumption of bandwidth, high transmission delay, and degradation of throughput which

affects the efficiency of congestion control [16]. However, efficient packet scheduling with congestion control is still in demand.

A. Motivation and Objectives

In this study, the transmission control protocol for cross-layer approach in IoT is designed for improving the performance of the network. This research also addresses the problems of high bandwidth consumption, high packet loss, high transmission latency, and low throughput to enhance TCP performance. The performance of TCP is affected by high packet loss, throughput degradation, high transmission latency, and bandwidth consumption. The existing works address these problems but, still, it does not provide a proper solution for TCP performance improvement. We are motivated by the major problems of this research which are listed below:

- Inappropriate Congestion Control: The existing works perform congestion control by considering RTT and RTO information; however, they did not estimate based on the current network status which leads to high packet loss and transmission delay. In addition, the count of duplicate acknowledgement (DACK) also affects the process of fast retransmission and fast recovery which degrades the throughput and Goodput.
- High Bandwidth Consumption: The existing work utilizes fixed RTO for congestion control mechanism which consumes high bandwidth when the packet transmission is completed before RTO expired; the completed packets need to wait until the RTO gets over. Further, the existing path selection methods consume high bandwidth as the initial data was sent over multiple paths in the network to find the optimal path.
- Inefficient Scheduling: In scheduling, most of the works only consider limited parameters (RTT, loss rate, and bandwidth) for packet scheduling and optimal path selection which provides inefficient scheduling which increases high jitter and packet loss. In addition, existing random subflow selection achieves high packet loss due to no consideration of completion time.

The above major gaps in the existing works motivated us to provide an effective objective in TCP. The major objective of this research is to reduce packet loss, and transmission latency and increase throughput and Goodput by performing packet scheduling and congestion control using cross-layer approach [17]. The other sub-objective of this research is listed as follows,

- To reduce jitter and transmission latency by performing efficient packet scheduling by considering several metrics such as transmission rate, queue length, completion time, etc. which improves the performance of congestion control.
- To increase the performance of TCP by performing cross layer-based congestion control in which the RTO is estimated by considering the current status of the network which increases the performance of congestion control.
- To increase the performance of fast retransmission and fast recovery by reducing the count of DACK that reduces transmission latency during data transmission.

B. Research Contribution

The proposed CoCoA3^{+}-TCP approach introduced cross-layer framework to improve the performance of TCP in IoT environment. The major contributions of this research are sorted as follows,

- Firstly, network construction is performed based on grid topology to increase the flexibility and scalability of the network which leads to high communication efficiency with low transmission delay.
- Secondly, packet scheduling is performed by implementing fitness based proportional fair (FPF) scheduling algorithm and selecting of best subflows and optimal path to reduce the jitter and transmission delay efficiently.
- Finally, congestion control in TCP is performed using CoCoA3^{+}-TCP algorithm by congestion detection and avoidance, fast retransmission, and recovery to improve the throughput and goodput which reduces the packet loss.

The performance of the proposed CoCoA3^{+}-TCP method is evaluated by considering several performance metrics to illustrate the performance of TCP such as throughput, goodput, packet loss, transmission delay, jitter, and congestion window size.

C. Paper Organization

The rest of the paper is organized as follows, Section II provides the existing works and its research gaps, section III emphasizes some specific problems in this field, and corresponding solutions are also provided. Section IV illustrates the proposed work with detailed explanation, relevant diagrams, equations, and pseudocodes. Section V explains the experimental analysis which consists of simulation setup, comparative analysis, and summary of the research are provided. Section VI tells about the conclusion and possible future direction of the proposed method.

II. Literature Survey

This section represents the literature survey of the previous approaches to increase the performance of TCP in terms of packet scheduling and congestion control. In addition, the research gaps in the previous methods are also described in this section.

A. Packet Scheduling Schemes

This sub-section describes the related works which were proposed for performing packet scheduling to increase the performance of TCP.

Authors in [18], proposed novel congestion control method for multipath TCP. The proposed work performed congestion control and scheduling for improving the performance of TCP.

This research used explicit congestion notification method for detection of shared bottleneck, which includes two processes such as subflow monitoring noticing the segment of TCP and estimating the congestion degree, and verification of shared bottleneck. Based on the shared bottleneck, congestion control is performed. Then packet scheduling is performed by considering fastest subflow, which is arranged based on RTT. Finally, simulation result shows that the proposed work achieved better performance in terms of throughput and detection accuracy. For subflow scheduling, this research only considers RTT which is not enough for optimal subflow scheduling leading to inefficient scheduling, therefore it degrades the performance of multipath TCP.

In [19], authors proposed an efficient packet scheduling method for improving the performance of multipath TCP. The proposed scheduling algorithm computes the number of packets is sent for every path. For that, optimal path is selected by considering buffer rate and completion time, and RTT. In addition, congestion control is performed based on the size of congestion window. For every path selection, congestion window size is evaluated and updated. The experimental results demonstrate that the proposed work achieved superior performance compare to existing approaches. Here, optimal path is selected by considering limited parameters which are not enough for selecting optimal path; hence it leads to degradation of throughput and Goodput.

Authors proposed coupled bottleneck bandwidth and round-trip propagation time (BBR) for improving the performance of multipath TCP by performing congestion control in [20]. The proposed work includes two phases such as congestion control and adaptive scheduling. Initially, coupled BBR evaluates the bandwidth for allocating the sending rate for every subflow for performing congestion control. Then scheduler selects the optimal path by considering the RTT and bandwidth to increase reliability and robustness. Here, packets are scheduled using pre-scheduling algorithm. The experimental result shows that the proposed work achieved better performance in terms of throughput and Goodput. Here, packet scheduling is performed by using pre-scheduling algorithm. However, it considers only limited parameters for scheduling which is not enough for optimal scheduling, in addition, random subflow based scheduling is performed which leads to high transmission latency and low throughput.

Authors proposed an approach to perform congestion control based on cross-layer in the wireless sensor network using fuzzy sliding method for controlling the modes [21]. Initially, signal to noise ratio (SNR) was applied to the channel in the TCP model with cross-layer-based congestion control in the middle of MAC layer, and transmission layer. Then integrating the sliding-mode control with fuzzy control for designing the Fuzzy-Sliding Mode Controller (FSMC) manages the buffer queue length in the congested nodes adaptively. Finally, NS-2.35 was used to perform simulation for comparing the proposed approach with several state-of-the-art methods in terms of packet loss, throughput, convergence, and delay. Here congestion control was performed using FSMC. However, lack of considering the priority of packets leads to high packet loss.

In [22], authors proposed an approach to perform congestion control and scheduling in the wireless network dynamically for reducing the delay. Initially, scheduling of selected data flow was performed to minimize the end-to-end delay using virtual rate scheduling algorithm which modifies the scheduling scheme. Flow control was performed based on windows to reduce the buffer overflow. In scheduling, each slot is decoupled into various mini-slots for low complexity. In addition, the congestion control was effectively performed by separating the mini-slots into numerous micro slots. Adjustment of virtual rate was performed to identify the priority of every flow link.

Authors proposed multi-path scheduling and congestion control by implementing deep reinforcement learning approach [23]. Initially, packet scheduling and congestion control framework was proposed by implementing deep Q learning (DQL) for the multipath TCP. The DQL technique was used to perform congestion control and scheduling optimally by the intelligent agents based on experience using policy gradients. Dynamic scheduling and congestion control were performed across all network paths by integrating the DQL with policy gradients. Recurrent neural network and long-short term memory algorithm were used to learn the behavior of paths and adjusts the scheduling and congestion control appropriately. Finally, experimental analysis was performed by comparing the performance of this approach with existing multipath TCP algorithms.

B. Congestion Control Schemes

This sub-section describes the related works which were proposed for controlling the congestion to increase the performance of TCP.

A dynamic congestion window algorithm for IoT environment was proposed by the authors in [24]. The main objective of this research is to minimize the delay and maximize the packet delivery ratio. In this research, congestion window size is dynamically changed based on the transmission rate, and bandwidth. The congestion window is adjusted only when the paths need to transmit the data. The simulation results demonstrate that the proposed work achieved superior performance in terms of packet delivery ratio, delay, and throughput. Here, the congestion window is adjusted based on the transmission rate and bandwidth which is not enough for congestion control due to lack of significant parameters such as RTO, RTT, and queue status, hence it leads to poor congestion.

A novel congestion control algorithm to improve the performance of improved multipath TCP was proposed in [25]. The main aim of this research is to reduce the delay and increase the throughput. The proposed work includes four processes such as startup, drain, probe bandwidth, and probe RTT which estimate the size of the congestion window considering pacing rate, congestion window, and send quantum. In this way, congestion control is performed for improving the performance of multipath TCP. The simulation result shows that proposed work achieved better performance in terms of throughput compared to existing approaches.

A novel method to improve the performance of TCP was proposed in [26] by considering queue size, retransmission, medium access control (MAC) overhead, and node energy. Here, the contention window is adjusted based on residual energy, queue size, slot time, and congestion window. Scheduling is done by estimating backoff time in a dynamic manner which improves the performance of TCP. The experimental result shows that the proposed work achieved better performance in terms of throughput, overhead, and energy consumption.

The congestion control mechanism for multipath TCP based on the flow control of the data was proposed by the authors in [27]. The proposed congestion control algorithm includes two main goals; first one is enhancing the throughput compared to single-path TCP. Second one is collecting the subflows by shared bottleneck. The bottleneck detection is performed based on RTT, packet loss, and congestion notification. For that purpose, this research proposed three filters such as RTT filter, congestion notification filter, and packet loss filter. Based on these filters the proposed work performed congestion control for multipath TCP. The simulation result shows that the proposed work achieved better performance in terms of detection accuracy and latency. Here, congestion window is adjusted based on three parameters that perform well, but the values are not considered based on the current network status, hence it leads to performance degradation of multipath TCP.

Authors in [28], proposed a congestion control method to improve the performance of TCP. The proposed work used window-based congestion control for resolving the limitations of the TCP in wireless networks. Here, congestion control is performed by considering queue level, sending rate, and flow utilization rate. This research includes three processes such as congestion window growth, congestion control, and recovery. In which the window growth is estimated based on threshold value (i.e., maximum window size). Congestion control is performed by considering two-level notifications. Finally, stability of the throughput is estimated based on transmission rate, and utility rate. The simulation shows that the proposed work achieved superior performance in terms of throughput, end-to-end delay, and Goodput. Authors proposed an approach to perform multipath scheduling of packets for controlling the network congestion concerning buffer acknowledgment in [29]. This method consists of two major entities a multi-path-packet scheduler (MPS), and a multi-path-congestion controller (MCC) in which the MPS was used to schedule the path based on three delay probabilities such as transmission delay and In-Out delay, and congestion delay. The MCC was used to manage the congestion control by considering the transmission control rate with respect to buffer availability, and packet delivery rate for increasing the throughput with low packet loss. Finally, simulation was performed to prove the performance of this approach in terms of buffer utilization, and throughput. Here, path scheduling was performed to reduce the transmission delay. However, lack of considering the data priority leads to high data loss.

The related works from [18 to 23] represents the packet scheduling schemes of the state-of-the art works whereas, the works [24 to 29] describes the congestion control schemes of the previous approaches. These methods have several limitations which are described at the end of each previous method. These limitations pave the way for the proposed CoCoA3^{+}-TCP method.

III. Problem Statement

The retransmission-based effective TCP congestion control method for IoT was introduced in [30]. Bottle-neck bandwidth round trip propagation time improves the fairness of RTT for congestion control was introduced in [31]. The major problems employed in this research are,

- Authors performed RTT estimation to calculate RTO for exponential backoff to improve congestion control of TCP. However, fixed RTO increases the waiting time of packets which leads to high latency and bandwidth consumption.
- Authors considered only queue status and delivery rate for adjusting CWND to perform congestion control. However, these parameters are not enough to perform effective congestion control leading to high jitter and low Goodput.
- Authors performed retransmission after obtaining three duplicate ACKs for reducing packet delivery failure. However, it increases the total delay of transmission which degrades the performance of TCP.

The packet scheduling and congestion control framework based on BBR to enhance the fairness of multipath TCP wireless networks was introduced in [32]. The major problems employed in this research are:

- Authors considered RTT to adjust the CWND for the improvement of congestion control. However, lack of considering RTO that leads to degradation of throughput.
- Two or more subflows share the same bottleneck set simultaneously for congestion control. However, these subflows are selected randomly without considering buffer rate and data size which leads to high transmission delay and packet loss.
- Several parameters such as bottleneck bandwidth, RTT, loss rate, and subflows are considered for packet scheduling. However, these parameters are not enough for optimal scheduling of packets which leads to high transmission delay.

The cross-layer congestion control framework for IoT environment was introduced in [33]. The major problems employed in this research are:

- Authors selected paths by considering only path bandwidth for congestion avoidance. However, single parameter is not enough to select the optimal path that leads to high jitter and packet loss.
- The proposed congestion control policy is used for performing fast retransmission based on adaptive data rate with respect to the path. However, lack of

considering RTO leads to high bandwidth consumption and high retransmission delay.

A. Research Solutions

To overcome the above problems faced by the existing works, the proposed work initially constructed the network as grid topology based on Manhattan distance which increases the flexibility and scalability of the network. After that, efficient packet scheduling is performed to reduce the latency during transmission and jitter using fitness based proportional fair scheduling algorithm which calculates the fitness value for every scheduling slot. The subflow and optimal path selection is done after packet scheduling to improve the throughput. The congestion control is performed to improve the performance using CoCoA3^{+}-TCP Algorithm that consists of three sub-processes namely congestion detection and avoidance, fast retransmission, and fast recovery. Here, congestion is detected by several metrics in that RTO is adaptively estimated using CMO algorithm. After congestion is detected fast transmission and fast recovery are performed by reducing the count of duplicate acknowledgment which reduces the transmission latency during data transmission. The overall proposed work improves the throughput, and goodput, and reduces the jitter and transmission latency in the TCP IoT environment.

IV. Proposed Work

This section provides the research methodology of the proposed cross-layer CoCoA3^{+}-TCP. The proposed work mainly focuses on improving the performance of the TCP using the cross-layer approach in an IoT environment. The performance of TCP is affected by high packet loss, Round Trip Time (RTT), and maximum segment size which lead to high latency and low throughput. The proposed work uses cross-layer design which allows the interaction between the data link layer and transport layer to transfer the information. Fig. 1 denotes the overall architecture of the proposed work in which cross-layer approaches among two layers is depicted.

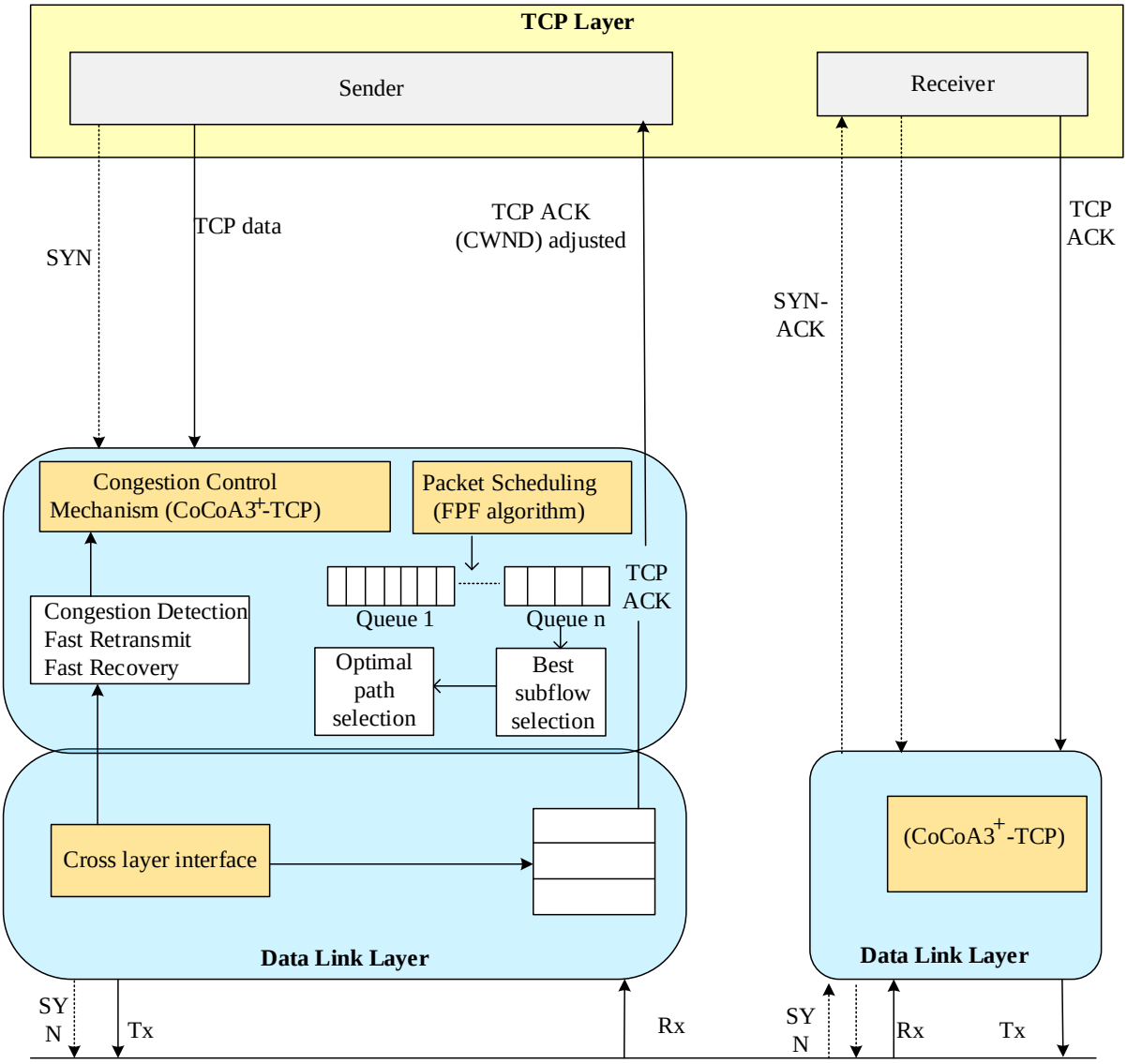

Fig. 1. CoCoA3^{+}-TCP Architecture.

Initially, the network is constructed by Manhattan distance-based grid topology which is a simpler grid topology that increases the flexibility and scalability of the network. The nodes are arranged in grid size of order 10×10. The proposed work assumes that the nodes in the network have stable range of communication range and that there are no crashes among the IoT nodes. Each node can communicate with each other, which are in the range of their respective communication. The range of transmission is constrained so that there is immediate communication among the nodes is established. Let (C^{n_i}, D^{n_i}) be the coordinates of node n_i, hence the distance (Dis) among the node n_i and node n_j can be formulated as.

$$Dis_{n_i,n_j} = |C^{n_i} - C^{n_j}| + |D^{n_i} - D^{n_j}| \quad (1)$$

The above equation denotes the arrangement of IoT nodes in the environment in 10×10 by Manhattan distance measure. This research improves the TCP performance by providing two processes which are listed as follows,

- Efficient Packet Scheduling
- TCP Congestion Control

A. Efficient Packet Scheduling

The main aim of packet scheduling is to reduce the jitter and transmission latency and improve the Goodput and throughput. Here, packets are scheduled and transmitted to the receiver which reduces transmission delay and congestion. For packet scheduling, we proposed FPF by considering bandwidth, energy (En), delay (dl), transmission rate (TXr), queue length (Ql), buffer rate (BR), completion time (cti) and data size (dz)which can be formulated as,

$$P^{\blacksquare} = arg\ \max_p \left(\frac{SR_{k,b}}{(W-1)T_k + \sum_{b=1}^{B} I_{k,b} SR_{k,b}} \right) \quad (2)$$

Where, $P^{\blacksquare}$ denotes the important value of each user which is used for effective packet scheduling, $SR_{k,b}$ rate of service for the k^{th} user at b^{th} time index based on the above-mentioned packet scheduling metrics, T_k indicates the $k-th$ user throughput, W is the window size, $I_{k,b}$ denotes the indicator variable which indicates the, when $I_{k,b} = 1$ the $k-th$ user packet is scheduled at $b-th$ time index else 0. The T_k can be formulated as.

$$T_k(t+1) = \begin{cases} \left(1 - \frac{1}{W}\right) T_{k,b}(t) + \frac{1}{W} A_k(t), V = P^{\blacksquare} \\ \left(1 - \frac{1}{W}\right) T_{k,b}(t),\ V \neq P^{\blacksquare} \end{cases} \quad (3)$$

Where, T_k denotes the past throughput that is dispensed to users in the previous transmission time interval. $T_k(t+1)$ denotes the current throughput which is calculated based on the T_k. V denotes the index of the user packets, A_k indicates the user throughput in current transmission time interval. Based on the importance value of each user fitness value is calculated. Here, fitness value is used to enhance long-term fairness and throughput in every scheduling slot which can be formulated as,

$$Fit(U) = \begin{cases} \max_{SR}(En, TXr, Ql) \\ \min_{SR}(dl, cti, dz, BR) \end{cases} \quad (4)$$

The above equation denotes the fitness value of each user packet. The user packets are scheduled based on the above-mentioned equation.

After completed packet scheduling best subflow is selected for scheduling which reduces the transmission delay during data transmission. The best subflow is selected based on the priority and fitness values of the user packets. The user packet with highest $Fit(U)$ are given more priority which can be formulated as

$$SF = \begin{cases} \uparrow\ Fit(U),\ SF_1 \\ \vdots \\ \downarrow\ Fit(U),\ SF_n \end{cases} \quad (5)$$

From the above equation, the $\uparrow\ Fit(U)$ indicates the user packet with higher fitness value who is given more priority and selected as first subflow while $\downarrow\ Fit(U)$ denotes the user packet with lower fitness value who is given less priority and selected as $n-th$ subflow. Finally, optimal path is selected from multiple paths for sending scheduled subflows. For optimal path selection, we consider congestion rate, transmission delay, and high throughput. Finally, TCP performances are improved by sending the scheduled subflow through optimal path. Fig. 2 represents the efficient packet scheduling, sub scheduling, and optimal path selection of the proposed work.

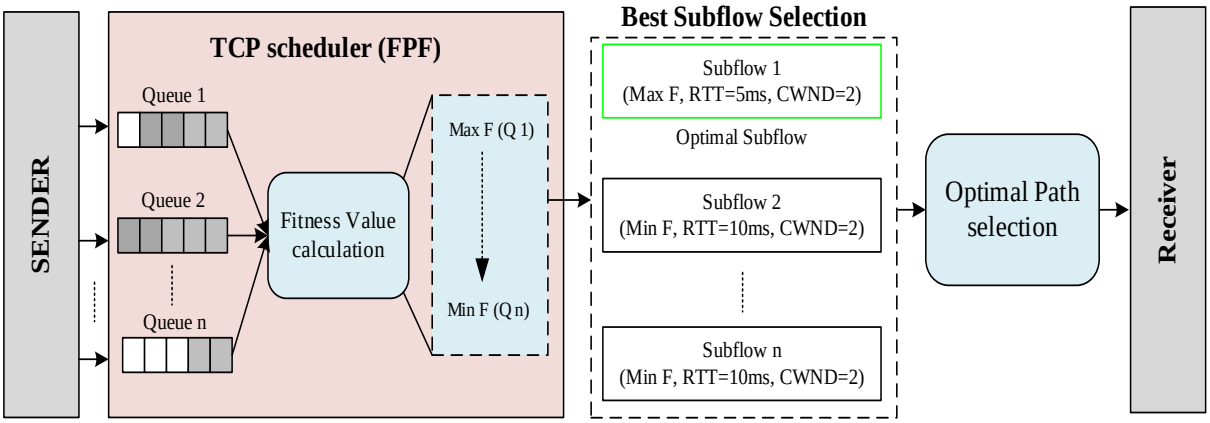

Fig. 2. FPF based Efficient Packet Scheduling.

B. TCP Congestion Control

In general, the network has congestion due to many packets being sent by the senders at a particular time, which affects the performance in terms of throughput, and packet delivery rate in TCP. To improve the performance of TCP, we need to avoid congestion in the network. The proposed congestion control CoCoA3^{+}-TCP Algorithm includes three sub-processes such as congestion detection and avoidance, fast retransmission, and fast recovery.

Initially, congestion is detected by considering RTT, Retransmission Timeout (RTO), transmission rate, delay, bandwidth, delivery rate, and queue status.

1) Round trip time (RTT): The amount of time taken for data to reach its destination and return the acknowledgment is known as RTT. The RTT can be formulated as,

$$RTT = 2 \times Pd \tag{6}$$

Where Pd denotes the propagation delay. The good protocol must have less Pd thereby having less RTT.

2) Transmission rate (TXr): The TXr is defined as the amount of data transmitted in a specific channel over a unit of time. In TCP the TXr is dependent on the size of the window which can be formulated as,

$$W = \frac{trf_t}{speed} \tag{7}$$

Where W is window size, generally TXr depends on W in TCP, trf_t indicates the time of transfer, and $speed$ denotes the speed of transfer, respectively.

3) Delay (dl): The delay is defined as the amount of time taken for data to transmit from source to destination. Generally, unit of delay depends on the time is taken nature of data. The delay must be less for a good communication protocol.

4) Bandwidth(BW): The BW is defined as the capacity of the network to withstand with high amount of data over a medium of transmission. The BW can be formulated as,

$$BW = \frac{W}{RTT} \tag{8}$$

Where W denotes the size of the window, and RTT denotes the round-trip time.

5) Delivery rate (DR): The amount of data packet delivered to the destination in a period is known as DR. Generally, a good TCP must have high DR.

6) Queue status (QS): The QS is defined based on the queue length. Higher the Queue length greater the congestion rate. The QS can be formulated as,

$$Ql = (1 - W) \times Ql + W \times Q_{sam} \tag{9}$$

Where, W is the window size, Q_{sam} is the sample queue that denotes the actual queue length of the packet arrival.

Here, the RTO value is estimated dynamically rather than fixed as a static value based on the current status of the environment. Then overall RTO value is estimated which is optimally calculated by CMO which is presented in CoCoA3^{+}-TCP. The CMO algorithm is a nature-inspired algorithm that mimics cat and mouse behaviours. This algorithm consists of two phases namely missing acknowledgment modelling and packet retransmission modelling. The population of packets in the TCP-IoT network is determined as,

$$R = \begin{bmatrix} R_1 \\ \vdots \\ R_i \\ \vdots \\ R_n \end{bmatrix}_{n \times m} = \begin{bmatrix} R_{1,1} & \cdots & R_{1,d} & \cdots & R_{1,m} \\ \vdots & & \vdots & & \vdots \\ R_{i,1} & & R_{i,d} & & R_{i,m} \\ \vdots & & \vdots & & \vdots \\ R_{n,1} & & R_{n,d} & & R_{n,m} \end{bmatrix}_{n \times m} \tag{10}$$

From the above equation, R denotes the packets population matrix, R_i denotes the exploration agent $(i - th)$, n indicates the number of packets transmitted in the network, and the problem variable is denoted as m. In the network, each transmitted packet must be objective function which can be formulated as,

$$OF = \begin{bmatrix} OF_1 \\ \vdots \\ OF_i \\ \vdots \\ OF_n \end{bmatrix}_{n \times 1} \tag{11}$$

Where, OF is the objective function vector, and OF_i is the exploration agent objective function. From the above OF values the best and worst transmitted packets are selected that can be formulated as,

$$OF_{alt} = \begin{bmatrix} OF_1^{alt} & Mini(OF) \\ \vdots & \vdots \\ OF_n^{alt} & Maxi(OF) \end{bmatrix}_{n \times 1} \tag{12}$$

From the above equation, OF_{alt} denotes the altered transmitted packet objective function based on equation (11). The transmitted packet with minimum objective function is considered as the best member else worst member. The overall packet population matrix consists of two groups namely missing acknowledgment packets and retransmitted packets. From that, the population of missing acknowledgment packets and retransmitted packets can be formulated as follows:

$$Mi(R) = \begin{bmatrix} Mi(R)_1 = R_1^{alt} \\ \vdots \\ Mi(R)_i = R_i^{alt} \\ \vdots \\ Mi(R)_{nR} = R_{nR}^{alt} \end{bmatrix}_{nR \times m} = \begin{bmatrix} R_{1,1}^{alt} & \cdots & R_{1,d}^{alt} & \cdots & R_{1,m}^{alt} \\ \vdots & & \vdots & & \vdots \\ R_{i,1}^{alt} & & R_{i,d}^{alt} & & R_{i,m}^{alt} \\ \vdots & & \vdots & & \vdots \\ R_{nR,1}^{alt} & & R_{nR,d}^{alt} & & R_{nR,m}^{alt} \end{bmatrix}_{nR \times m} \tag{13}$$

$$rt(R) = \begin{bmatrix} rt(R)_1 = R_{nR+1}^{alt} \\ \vdots \\ rt(R)_j = R_{nr+j}^{alt} \\ \vdots \\ rt(R)_{nrt} = R_{nR+nrt}^{alt} \end{bmatrix}_{nrt \times m} =$$

$$\begin{bmatrix} R_{nR+1,1}^{alt} & \cdots & R_{nR+1,d}^{alt} & \cdots & R_{nR+1,m}^{alt} \\ \vdots & & \vdots & & \vdots \\ R_{nR+j,1}^{alt} & & R_{nR+j,d}^{alt} & & R_{nR+j,m}^{alt} \\ \vdots & & \vdots & & \vdots \\ R_{nR+nrt,1}^{alt} & & R_{nR+nrt,d}^{alt} & & R_{nR+nrt,m}^{alt} \end{bmatrix}_{nrt \times m} \quad (14)$$

From the above equations (13) and (14), $Mi(R)$ denotes the missing acknowledge packets and $rt(R)$ denotes the retransmitted packets respectively. Therefore, in first phase, the missing acknowledgment packets are found by,

$$Mi(R)_i = \begin{cases} Mi(R)_i^{new}, |OF_i^{Mi(R),new} < OF_i^{Mi(R)} \\ Mi(R), else \end{cases} \quad (15)$$

Where, $Mi(R)_i^{new}$ is missing acknowledgment based on current network status, and $OF_i^{Mi(R),new}$ denotes the objective of missing acknowledgment. In second phase, the retransmitted packet is found by,

$$rt(R)_j = \begin{cases} rt(R)_i^{new}, |OF_i^{rt(R),new} < OF_i^{rt(R)} \\ rt(R), else \end{cases} \quad (16)$$

Where, $rt(R)_i^{new}$ new retransmitted packets based on current network status, and $OF_i^{rt(R),new}$ denote the objective function of new retransmitted packets. From the equation (15) and (16), the RTO is adaptively determined by,

$$RTO = Mi(R)_i^{new} - rt(R)_i^{new} \quad (17)$$

The pseudocode for adaptive RTO estimation based on cat and mouse optimization algorithm is given below:

Pseudocode
Adaptive RTO selection ()
Start CMO
Initialize R using (10)
Set exploration agents (n) and iterations (T)
Compute OF using (11)
Alter the OF using (12)
Set the $Mi(R)$ population using (13)
Set the $rt(R)$ population using (14)
Stage 1: Missing ACK stage
For i=1: nR;
Find the missing acknowledged packets using (15)
End
Stage 2: packet retransmission modelling
For j=1: nrt;
Find retransmitted packets using (16)
End
Calculate RTO using (17)
End

Due to congestion, the network has high packet loss; hence we need to perform fast retransmission and fast recovery. The basic goal of fast retransmission is to minimize the count of duplicate acknowledgment (DACK) which reduces the transmission latency during data transmission. The steps involved in fast retransmission are:

- Step 1: Sender sent some packets to the receiver, and receiver receives the packets and gives acknowledgment.
- Step 2: If correct acknowledgment, the sender sends another packet. Else, sender sends acknowledge message along with lost packets to sender.
- Step 3: Along with lost packets, sender continuously sends another packet which leads to two DACK.
- Step 4: After getting two DACK, the sender retransmits the missing packets (i.e., Fast recovery) without exceeding RTO based on the current network status which is evaluated by RTT for every packet given in equation (6).

The proposed algorithm adjusts the congestion window size that reduces the delay which is illustrated in Fig. 3. The current network status is determined based on RTT. If the RTT increases beyond network limit then there is congestion in the network, and when RTT gets decreased there is no congestion, at that time size of congestion window must be increased to avoid the congestion further.

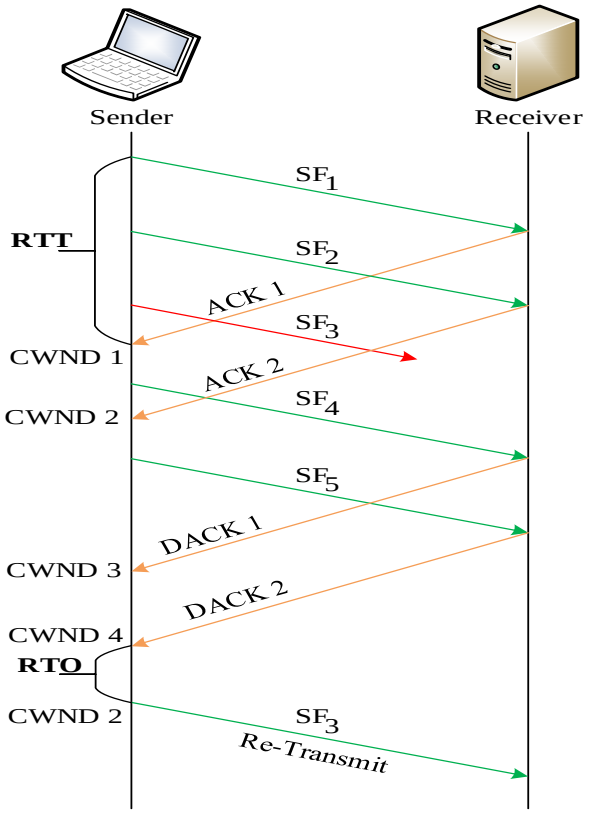

Fig. 3. CWND Adjustment for Fast Retransmission and Recovery.

V. Experimental Results

In this section, the proposed CoCoA3^{+}-TCP is experimented with and validated. This section is further subdivided into four sub-sections such as simulation setup, application scenario comparative analysis, and research summary.

A. Simulation Setup

This sub-section describes the simulation of the proposed work. Initially, the network is constructed as grid topology based on Manhattan distance for improving the scalability of the network. This work utilizes the system with hardware configuration of random-access memory (RAM) 8GB, hard disk 256GB, and processor of Intel(R) Core (TM) i5-3210M CPU @ 2.50GHz; the software configuration of network simulator with version 3.26, and operating system of Ubuntu Linux 18.04. Table I denotes the various simulation parameters used for simulating the proposed model. Fig. 4 represents the simulation grid topology of the proposed CoCoA3^{+}-TCP.

The above figure represents the simulation grid topology of the proposed CoCoA3^{+}-TCP. The proposed work constructs the grid as size of 10 × 10 using Manhattan distance which is flexible and mitigates the scalability issues. By this grid topology construction, the proposed work achieves high transmission efficiency, high throughput and less delay during packet scheduling and congestion control, respectively.

TABLE I. Simulation Configuration

Simulation Parameter	Values
Network Simulator	NS-3.26
Routing Protocol	AODV
Type of Traffic	TCP
Area of Simulation	1500m × 1500m
No. of IoT nodes	100
No. of Base station	1
Size of the window	2^8
Size of the packet	2000 bytes
Size of the Queue	30 packets
Placement of IoT node	Grid topology
Topology Size	10 × 10
Time for Total Simulation	300ms

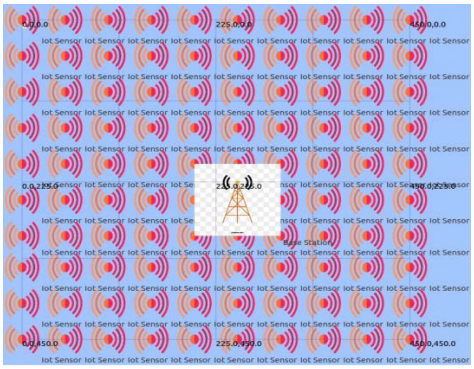

Fig. 4. Simulation Grid Topology of CoCoA3^{+}-TCP.

B. Application Scenario

The proposed TCP-IoT model can be adopted in medical environment (i.e. patient monitoring system). The medical environment consists of numerous patients and each patient are equipped with IoT sensors (i.e. heart beat sensor, blood pressure sensor, etc.). The improper counter measure in the medical environment leads to severe threats to the patients. To take correct medical counter measure at correct time period, the sensed data are optimally report to the doctors. For that we propose, reliable TCP-IP congestion control fast and reliable mechanism in which every sensor data is forwarded to the server where it performs scheduling using FPF algorithm based on several metrics such as bandwidth, delay, buffer rate, etc. The scheduled data are selected best sub flow using CoCoA3^{+}-TCP algorithm. The proposed work can be applied in the medical environment for patient monitoring reduces the latency, and congestion issues.

C. Comparative Analysis

In this sub-section, the comparative analysis is performed for the proposed CoCoA3^{+}-TCP method with several previous approaches such as ICC-TCP (28), and BC-CPS (30) methods for evaluating the performance of these methods. Various performance metrics are considered in terms of throughput, goodput, packet loss, transmission delay, jitter, and congestion window size respectively to evaluate the performance of the proposed CoCoA3^{+}-TCP method and other previous works.

1) Impact of throughput: This metric is used to evaluate the total number of data packets received ($\dot{p}$) at a certain period ($\dot{t}$) by the receiver. Throughput ($\dot{c}$) is denoted in terms of bits per second. The formulation of this metric is described as follows,

$$\dot{c} = \frac{\dot{p}}{\dot{t}} \tag{18}$$

Fig. 5 illustrates the comparison of throughput between the proposed CoCoA3^{+}-TCP method with several previous methods such as ICC-TCP, and BC-CPS methods with respect to the packet size. A network with high throughput achieves better communication between the source and destination. Throughput increases with increasing the packet size.

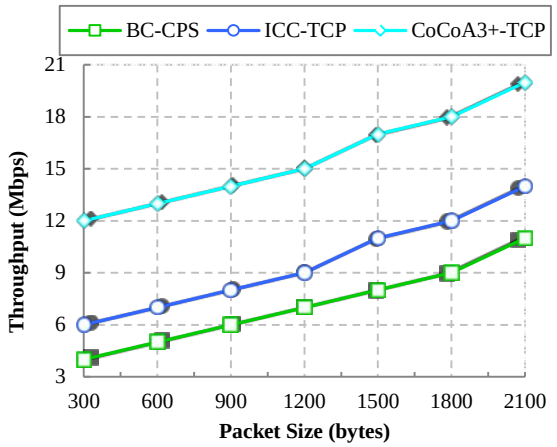

Fig. 5. Throughput Vs. Packet Size.

In the existing BC-CPS approach, congestion control and packet scheduling are statically performed with poor RTO measurement, and retransmission was performed after several ACK which increases the latency that leads to low throughput. In the proposed CoCoA3^{+}-TCP method, adaptive RTO estimation is performed and retransmission of packets is performed after receiving two DACK which reduces the latency that increases the throughput of the proposed CoCoA3^{+}-TCP method. The comparative results show that the proposed CoCoA3^{+}-TCP method achieves high throughput (12-20 Mbps) when compared with ICC-TCP (6-14 Mbps) method and BC-CPS (4-11 Mbps) method.

Similarly, Fig. 6 shows the comparison of throughput with respect to simulation time. From this figure, it is clearly shown that the proposed CoCoA3^{+}-TCP method achieves high throughput of 20 Mbps for the packet size of 2100 bytes which is 6 Mbps higher than the ICC-TCP method and 9 Mbps higher than BC-CPS method.

2) Impact of goodput: It is one of the important metrics which is the other form of throughput in the application level to evaluate the performance of the proposed algorithm for performing accurate and efficient communication, especially during huge data transmission. Goodput (ß) is formulated by the ratio between the amount of particular data (ɯ) to the time taken for reaching the receiver (ɧ) which is expressed as follows:

$$ß = \frac{ɯ}{ɧ} \tag{19}$$

Fig. 7 represents the goodput comparison for the proposed CoCoA3^{+}-TCP method and several previous works in terms of packet size. A network with low transmission delay achieves high goodput which increases the communication efficiency. The goodput increases with respect to increase of packet size. In the existing BC-CPS method, lack of considering the RTO when evaluating the RTT for the communication increases the latency which reduces the performance of the TCP by attaining low goodput.

In the proposed CoCoA3^{+}-TCP method, the RTO is estimated based on current status of the network which increases the TCP performance by achieving high goodput. From the graphical results, it is proved that the proposed CoCoA3^{+}-TCP method achieves high goodput when compared with state-of-the-art methods. The proposed CoCoA3^{+}-TCP attains high goodput of 800 Kbps at 2100 bytes. The difference of goodput between the proposed CoCoA3^{+}-TCP method and the ICC-TCP method is 120 Kbps, and 200 Kbps for the BC-CPS method.

Similarly, Fig. 8 represents the comparison of goodput to the simulation time. The comparative results show that the proposed CoCoA3^{+}-TCP method achieves high goodput of 800 Kbps which is 140 Kbps higher than ICC-TCP method and 200 Kbps higher than BC-CPS method at 300ms.

3) Impact of packet loss: It is used in the network to evaluate communication reliability. Packet loss is defined as the ratio of the number of data packets that are not received to the total amount of packets which is formulated as follows:

$$ṁ = \frac{ṫ}{∂} \tag{20}$$

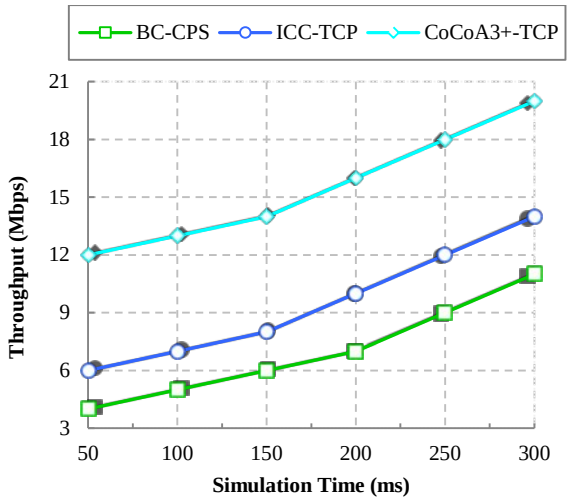

Fig. 6. Throughput Vs. Simulation Time.

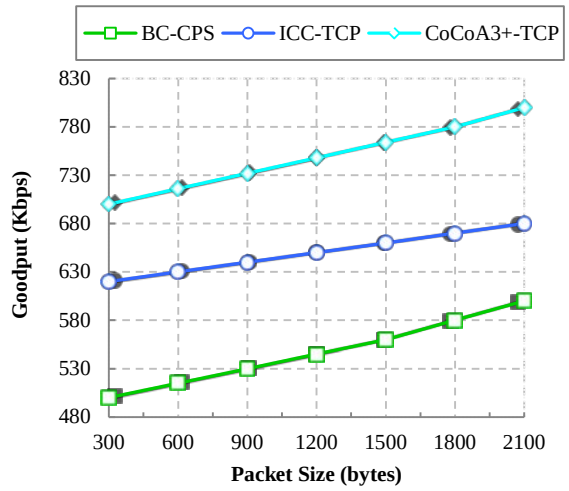

Fig. 7. Goodput Vs. Packet Size.

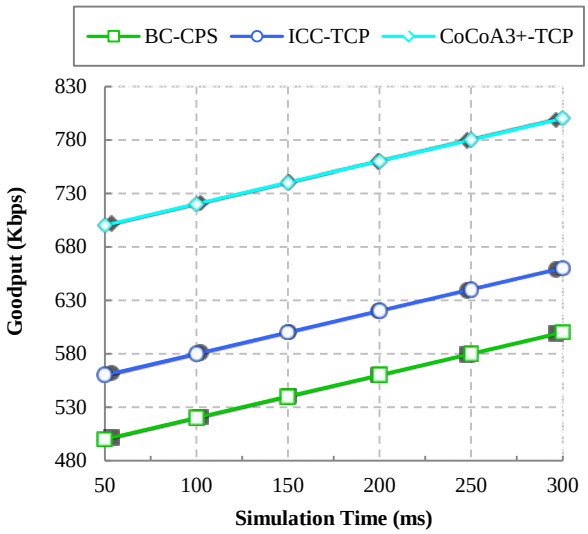

Fig. 8. Goodput Vs. Simulation Time.

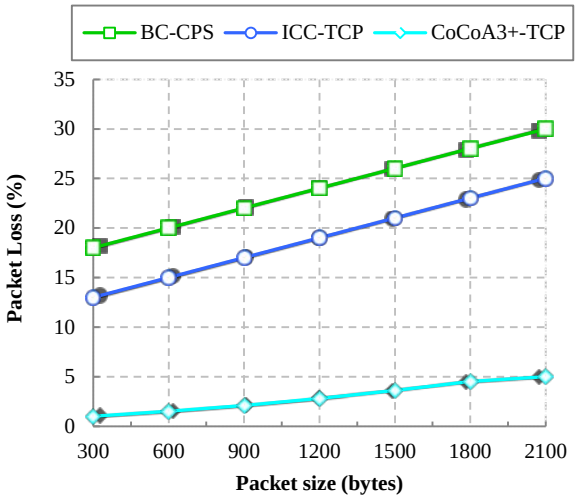

Fig. 9. Packet Loss Vs. Packet Size.

Where ⋔ shows the packet loss, ϯ represents the non-received data packets and ∂ denotes the total amount of packets. Fig. 9 shows the comparison of packet loss between the proposed CoCoA3^{+}-TCP method and several previous approaches. A network with low packet loss reduces the number of retransmissions which decreases the network congestion. The packet loss increases with increase in packet size. In the ICC-TCP methods, the congestion control is performed by evaluating the RTT without considering the present network status that increases the DACK which leads to high packet loss. In the proposed method, congestion control is performed by adaptively implementing CoCoA3^{+}-TCP algorithm. In addition, the congestion window is increased by one when more than two DACK reduces the packet loss.

The comparative results illustrate that the proposed CoCoA3^{+}-TCP method achieves low packet loss when compared with other previous methods. The proposed CoCoA3^{+}-TCP method achieves low packet loss of about 5% for 2100 bytes of packet size which is 20% lower than ICC-TCP method, and 25% lower than BC-CPS method. Table II describes the packet loss variation of the proposed CoCoA3^{+}-TCP method and other existing approaches.

4) Impact of transmission delay: Transmission delay (ɟ) represents the amount of additional time taken by the proposed CoCoA3^{+}-TCP method to transmit the packets from the source to the destination. It is formulated by the difference between the actual packet received time (ɵ) and the expected time (ζ) which is expressed as follows:

$$ɟ = ɵ - \zeta \quad (21)$$

TABLE II. NUMERICAL ANALYSIS OF PACKET LOSS

Methods	Packet Size
BC-CPS	24.3 ± 0.5
ICC-TCP	19.5 ± 0.4
CoCoA3^{+}-TCP	2.92 ± 0.1

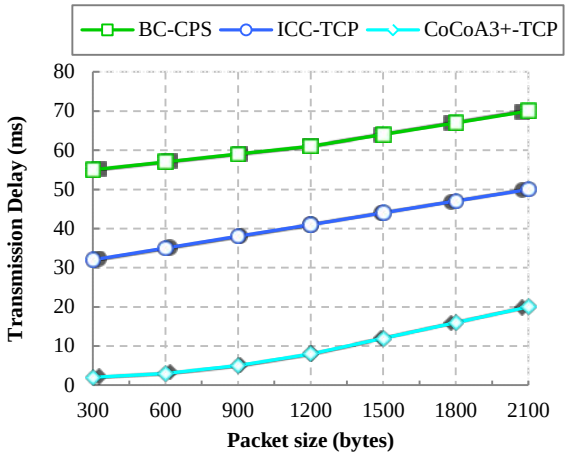

Fig. 10. Transmission Delay Vs. Packet Size.

Fig. 10 illustrates the comparison of transmission delay for the proposed CoCoA3^{+}-TCP method and other existing methods in terms of packet size. A network with low transmission delay increases the throughput as well as goodput. The transmission delay increases with increasing the packet size. In the BC-CPS method, random selection of subflows is performed with insufficient parameters such as RTT, loss rate, etc., for data transmission which increases the transmission delay and also leads to high packet loss. In the proposed CoCoA3^{+}-TCP work, Optimal selection of subflows is performed by considering buffer rate, queue length, etc. which improves the fairness for long term that reduces the transmission delay when compared with other previous methods. From this figure, it is clearly shown that the proposed CoCoA3^{+}-TCP method achieves low transmission delay of about 20ms for 2100 bytes of packet size which is 30ms faster than ICC-TCP method and 50ms faster than BC-CPS method.

Similarly, Fig. 11 shows the comparison of transmission delay with respect to simulation time between the proposed CoCoA3^{+}-TCP method and existing methods. The results show that the proposed CoCoA3^{+}-TCP method achieves low transmission delay of about 20ms for 300ms of simulation time which is 28ms faster than ICC-TCP method and 48ms faster than BC-CPS method.

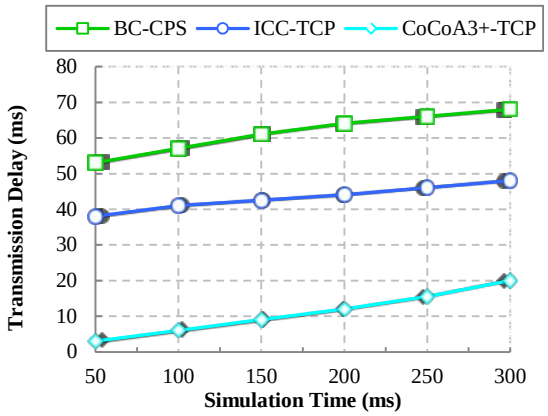

Fig. 11. Transmission Delay Vs. Simulation Time.

5) Impact of jitter: This metric is used to evaluate the difference between the transmitting time and receiving time. The formulation of jitter is defined as the time variation between the current and previous data packets which is expressed as follows:

$$\lambda = \xi - \varkappa \quad (22)$$

Where λ denotes the jitter, ξ represents the current packet receiving time, $\varkappa$ shows the previous packet receiving time.

Fig. 12 shows the comparison of jitter between the proposed CoCoA3$^+$-TCP method and several state-of-the-art approaches in terms of packet size. A network with low jitter increases the performance of TCP. The jitter increases with increasing the packet size. In the ICC-TCP method, DACK was not considered during retransmission which increases the latency of data recovery that leads to high jitter. In addition, lack of considering RTO in the BC-CPS method during congestion control also increases the jitter.

In the proposed CoCoA3$^+$-TCP method, fast data recovery is performed by considering the DACK more than two and reducing the congestion window to half of the present status of congestion window with respect to RTT and RTO is considered based on the network's current status that reduces the jitter effectively when compared with other previous works. The graphical results illustrate that the proposed CoCoA3$^+$-TCP method achieves low jitter of about 15ms for 2100 bytes of packet size. Whereas, the ICC-TCP and BC-CPS methods achieve jitter of about 50ms and 60ms for the same packet size which is 35ms and 45ms greater than the proposed CoCoA3$^+$-TCP method.

Similarly, Fig. 13 illustrates the comparison of jitter between the proposed CoCoA3$^+$-TCP method and other previous methods with respect to simulation time. The results show that the proposed CoCoA3$^+$-TCP method achieves low jitter of about 15ms which is 32ms faster than ICC-TCP method and 43ms faster than BC-CPS method.

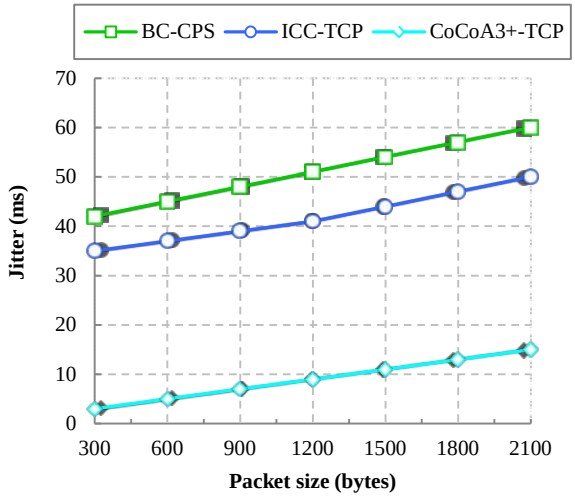

Fig. 12. Jitter Vs. Packet Size.

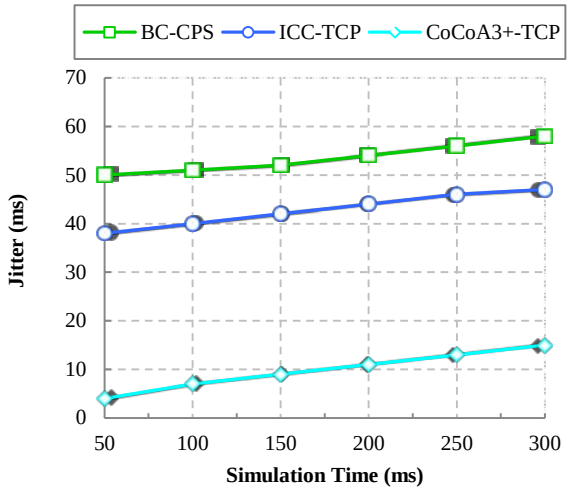

Fig. 13. Jitter Vs. Simulation Time.

6) Impact of congestion window size: This metric is used to evaluate the number of data packets transmitted from the source to the destination without any traffic. A network with large congestion window size increases the throughput of the network. Whenever DACK occurs, the size of the congestion window decreases to half of its size to perform fast retransmission and recovery.

Fig. 14 represents the comparison of congestion window size for the proposed CoCoA3$^+$-TCP method and other existing methods with respect to simulation time. The size of the congestion window increases when normal data transmission and decreases when retransmission occurs with increasing simulation time. In the state-of-the-art approaches, the size of the congestion window is randomly changed to send the data packets with lack of considering the RTO which increases the congestion that leads to high complexity in congestion control.

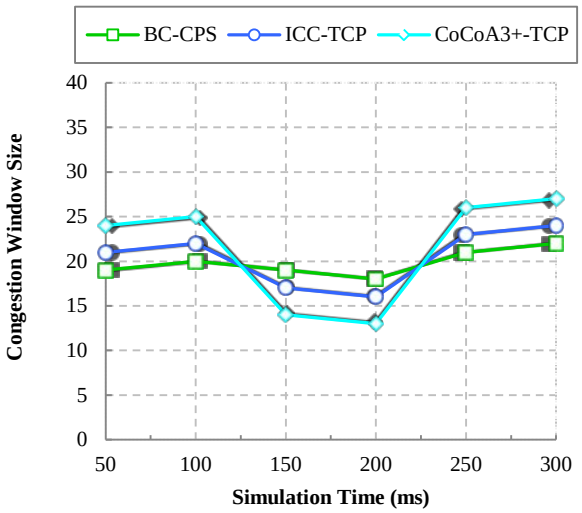

Fig. 14. Congestion Window Size Vs. Simulation Time.

In the proposed CoCoA3$^+$-TCP method, the congestion window is adaptively changed, i.e. whenever the DACK increased, the congestion window is increased by one. For faster data recovery, the congestion window is reduced by half with respect to RTT which reduces the congestion and increases the control of congestion. In addition, estimation of dynamic RTO is performed based on the network's current status which also reduces the retransmission of packets. The comparative results illustrate that the proposed CoCoA3$^+$-TCP method achieves better congestion window size when compared with other previous methods.

The proposed CoCoA3$^+$-TCP method attains high congestion window size of about 27 which is 4 larger than ICC-TCP method and 6 larger than the BC-CPS method. When retransmission occurs, the size of the congestion window decreases by about 13 which is 5 lower than BC-CPS method and 3 lower than ICC-TCP method. Table III illustrates the changes in congestion window size between the proposed CoCoA3$^+$-TCP method and several previous works.

TABLE III. NUMERICAL ANALYSIS OF CONGESTION WINDOW SIZE

Methods	**Simulation Time**
BC-CPS	19.3 ± 0.5
ICC-TCP	20.5 ± 0.3
CoCoA3$^+$-TCP	21.5 ± 0.1

D. Research Summary

This sub-section summarizes the overall performance of the proposed CoCoA3$^+$-TCP method in terms of throughput, goodput, packet loss, transmission delay, jitter, and congestion window size respectively from Fig. 5 to 14. The major highlights of the proposed CoCoA3$^+$-TCP method are described in this sub-section. Fitness based Proportional Fair (FPF) scheduling algorithm is proposed for performing efficient packet scheduling which improves throughput and Goodput by reducing the transmission delay and jitter. The CoCoA3$^+$-TCP algorithm is proposed for improving congestion control and avoidance with CMO by considering multiple parameters such as delay, transmission rate, RTT, RTO, and bandwidth. DACK count is minimized by performing fast retransmission and recovery which retransmits the packets before the threshold of RTO depending upon the current network status. Table IV represents the ablation study between the existing and proposed approaches which consists of the average values of the corresponding performance metrics. From this table, it is proved that the proposed CoCoA3$^+$-TCP method achieved high TCP performance when compared with other existing works.

TABLE IV. NUMERICAL ANALYSIS OF PROPOSED AND EXISTING METHODS

Performance Metrics		**BC-CPS**	**ICC-TCP**	**CoCoA3$^+$-TCP**
Throughput (Mbps)	Packet Size (bytes)	7.14 ± 0.5	9.58 ± 0.3	15.57 ± 0.1
	Simulation Time (ms)	7.15 ± 0.4	9.5 ± 0.2	15.5 ± 0.1
Goodput (Kbps)	Packet Size (bytes)	547.14 ± 0.5	650.20 ± 0.4	748.57 ± 0.1
	Simulation Time (ms)	550.20 ± 0.4	610.15 ± 0.3	750.18 ± 0.1
Transmission Delay (ms)	Packet Size (bytes)	61.85 ± 0.4	41.23 ± 0.2	9.42 ± 0.1
	Simulation Time (ms)	61.5 ± 0.5	43.25 ± 0.4	10.91 ± 0.1
Jitter (ms)	Packet Size (bytes)	51.5 ± 0.5	41.85 ± 0.2	9.3 ± 0.1
	Simulation Time (ms)	53.5 ± 0.4	42.83 ± 0.3	9.83 ± 0.1

VI. CONCLUSION AND FUTURE WORK

The cross-layer approach is proposed in this research to perform efficient packet scheduling and congestion control for improving the TCP performance in IoT networks. Initially, network construction is performed which improves the flexibility and scalability of the network. After network construction, packet scheduling is performed by considering numerous parameters using FPF scheduling algorithm, and best subflow is selected which reduces the transmission delay and jitter. Optimal path selection is performed by considering several parameters to increase the throughput and goodput. The congestion control in TCP is performed by proposing CoCoA3$^+$-TCP algorithm which consists of three sub-stages such as congestion control, fast retransmission, and fast recovery to reduce the transmission delay and improve the network performance. Fast retransmission is achieved by estimating the RTO dynamically using a cat and mouse based optimization (CMO) algorithm and fast recovery is performed by reducing the congestion window. The simulation is performed by network simulator (NS-3.26) and the results illustrate that the proposed CoCoA3$^+$-TCP method achieved high performance of TCP in IoT network when compared with other previous works.

In future, this research will be improved by providing security for TCP protocol in IoT environment with low complexity that reduces the transmission delay. In addition, the security increases the throughput and goodput with low retransmission.

REFERENCES

[1] V. Vijaya Kumar et al., "Design of peer-to-peer protocol with sensible and secure IoT communication for future internet architecture," Microprocess. Microsyst., vol. 78, no. February, p. 103216, 2020.

[2] N. Nikolov, "Research of MQTT, CoAP, HTTP and XMPP IoT Communication protocols for Embedded Systems," 2020 29th Int. Sci. Conf. Electron. 2020 - Proc., pp. 18–21, 2020.

[3] S. Parween, S. Z. Hussain, and M. A. Hussain, "A Survey on Issues and Possible Solutions of Cross-Layer Design in Internet of Things," Int. J. Comput. Networks Appl., vol. 8, no. 4, p. 311, 2021.

[4] M. Prakash and A. Abdrabou, "Performance Insights on Using Multipath TCP for Wireless Multihomed IoT Gateways," Proc. 2020 Int. Conf. Comput. Inf. Telecommun. Syst. CITS 2020, pp. 1–5, 2020.

[5] S. Z. Hussain and S. Parween, "Comparative Study of TCP Congestion Control Algorithm in IoT," In 2021 3rd International Conference on Advances in Computing, Communication Control and Networking (ICAC3N), pp. 1428–1431, IEEE, 2022.

[6] J. Wirges and U. Dettmar, "Performance of TCP and UDP over narrowband internet of things (NB-IoT)," Proc. - 2019 IEEE Int. Conf. Internet Things Intell. Syst. IoTaIS 2019, pp. 5–11, 2019.

[7] Y. Yang and L. Hanzo, "Permutation-Based TCP and UDP Transmissions to Improve Goodput and Latency in the Internet of Things," IEEE Internet Things J., vol. 8, no. 18, pp. 14276–14286, 2021.

[8] M. A. Alrshah, M. A. Al-Maqri, and M. Othman, "Elastic-TCP: Flexible Congestion Control Algorithm to Adapt for High-BDP Networks," IEEE Syst. J., vol. 13, no. 2, pp. 1336–1346, 2019.

[9] K. Miyazawa, S. Yamaguchi, and A. Kobayashi, "A study on cyclic performance fluctuation of CUBIC TCP and TCP BBR considering estimated RTT and bandwidth," Proc. - 2019 7th Int. Symp. Comput. Netw. Work. CANDARW 2019, pp. 478–480, 2019.

[10] S. Parween and S. Z. Hussain, "A Comparative analysis of CoAP based Congestion Control in IoT," pp. 321–324, 2021.

[11] S. Kishimoto, S. Osada, Y. Tarutani, Y. Fukushima, and T. Yokohira, "A TCP Incast Avoidance Method Based on Retransmission Requests from a Client," ICTC 2019 - 10th Int. Conf. ICT Converg. ICT Converg. Lead. Auton. Futur., pp. 153–158, 2019.

[12] G. Luan and N. C. Beaulieu, "Accurate mathematical modeling and solution of TCP congestion window size distribution," Comput. Commun., vol. 163, pp. 195–201, 2020.

[13] H. M. Noman, A. A. Abdulrazzaq, M. M. Kareem, and A. H. Ali, "Improvement Ivestigation of the TCP Algorithms with Avoiding Network Congestion Based on OPNET," IOP Conf. Ser. Mater. Sci. Eng., vol. 518, no. 5, 2019.

[14] P. Dong, J. Xie, W. Tang, N. Xiong, H. Zhong, and A. V. Vasilakos, "Performance Evaluation of Multipath TCP Scheduling Algorithms," IEEE Access, vol. 7, pp. 29818–29825, 2019.

[15] A. Marin, S. Rossi, and C. Zen, "Size-based scheduling for TCP flows: Implementation and performance evaluation," Comput. Networks, vol. 183, no. September, p. 107574, 2020.

[16] V. Adarsh, P. Schmitt, and E. Belding, "MPTCP performance over heterogenous subpaths," Proc. - Int. Conf. Comput. Commun. Networks, ICCCN, vol. July, 2019.

[17] S. Parween and S. Z. Hussain, "A review on cross-layer design approach in WSN by different techniques," Adv. Sci. Technol. Eng. Syst., vol. 5, no. 4, pp. 741–754, 2020.

[18] W. Wei, K. Xue, J. Han, D. S. L. Wei, and P. Hong, "Shared Bottleneck-Based Congestion Control and Packet Scheduling for Multipath TCP," IEEE/ACM Trans. Netw., vol. 28, no. 2, pp. 653–666, 2020.

[19] R. K. Chaturvedi and S. Chand, "An Adaptive and Efficient Packet Scheduler for Multipath TCP," Iran. J. Sci. Technol. - Trans. Electr. Eng., vol. 8, 2020.

[20] J. Han et al., "Leveraging coupled BBR and adaptive packet scheduling to boost MPTCP," IEEE Trans. Wirel. Commun., vol. 20, no. 11, pp. 7555–7567, 2021.

[21] S. Qu, L. Zhao, and Z. Xiong, "Cross-layer congestion control of wireless sensor networks based on fuzzy sliding mode control," Neural Comput. Appl., vol. 32, no. 17, pp. 13505–13520, 2020.

[22] K. Malarvizhi and L. S. Jayashree, "Dynamic scheduling and congestion control for minimizing delay in multihop wireless networks," J. Ambient Intell. Humaniz. Comput., vol. 12, no. 3, pp. 3949–3957, 2021.

[23] S. R. Pokhrel and A. Walid, "Learning to Harness Bandwidth with Multipath Congestion Control and Scheduling," IEEE Trans. Mob. Comput., vol. 1233, no. c, pp. 1–14, 2021.

[24] R. Chappala, C. Anuradha, and P. S. R. C. Murthy, "Adaptive Congestion Window Algorithm for the Internet of Things Enabled Networks," Int. J. Adv. Comput. Sci. Appl., vol. 12, no. 2, pp. 105–111, 2021.

[25] I. Mahmud, T. Lubna, Y. J. Song, and Y. Z. Cho, "Coupled multipath BBR (c-MPBBR): A efficient congestion control algorithm for multipath TCP," IEEE Access, vol. 8, pp. 165497–165511, 2020.

[26] R. M. Bhavadharini, S. Karthik, N. Karthikeyan, and A. Paul, "Wireless Networking Performance in IoT Using Adaptive Contention Window," Wirel. Commun. Mob. Comput., vol. 2018.

[27] I. Mahmud, T. Lubna, G. H. Kim, and Y. Z. Cho, "Ba-mpcubic: Bottleneck-aware multipath cubic for multipath-tcp," Sensors, vol. 21, no. 18, 2021.

[28] M. Joseph Auxilius Jude, V. C. Diniesh, and M. Shivaranjani, "Throughput stability and flow fairness enhancement of TCP traffic in multi-hop wireless networks," Wirel. Networks, vol. 26, no. 6, pp. 4689–4704, 2020.

[29] A. Kumar, P. V. S. Srinivas, and A. Govardhan, "A Multipath Packet Scheduling Approach based on Buffer Acknowledgement for Congestion Control," Procedia Comput. Sci., vol. 171, pp. 2137–2146, 2020.

[30] C. Lim, "Improving congestion control of TCP for constrained IoT networks," Sensors (Switzerland), vol. 20, no. 17, pp. 1–16, 2020.

[31] W. Pan, H. Tan, X. Li, and X. Li, "Improved RTT fairness of BBR congestion control algorithm based on adaptive congestion window," Electron., vol. 10, no. 5, pp. 1–18, 2021.

[32] W. Wei, K. Xue, J. Han, Y. Xing, D. S. L. Wei, and P. Hong, "BBR-Based Congestion Control and Packet Scheduling for Bottleneck Fairness Considered Multipath TCP in Heterogeneous Wireless Networks," IEEE Trans. Veh. Technol., vol. 70, no. 1, pp. 914–927, 2021.

[33] L. P. Verma and M. Kumar, "An IoT based Congestion Control Algorithm," Internet of Things, vol. 9, p. 100157, 2020.