Wireless Networks (2021) 27:4189–4198
https://doi.org/10.1007/s11276-021-02701-3

ORIGINAL PAPER

Reduce delay of multipath TCP in IoT networks

Mohammed Aljubayri[1] · Tong Peng[1] · Mohammad Shikh-Bahaei[1]

Accepted: 21 June 2021 / Published online: 12 July 2021

Abstract

Multi-homed devices such as smartphones, tablets and laptops are enabled with multiple heterogeneous interfaces available for transmission. Those interfaces can be utilized for simultaneous transmission of a single TCP flow using Multipath TCP (MPTCP). MPTCP is a protocol that is designed to increase end-to-end throughput and reliability of communications by splitting data through multiple parallel paths. Although delay in MPTCP enhanced significantly in the recent years, high number of data transmissions remains an issue. In this paper, we reduce MPTCP delay by reducing the number of transmissions using Opportunistic Routing (OR) technique. OR is a routing model used to increase the delivery rate and reliability of data transmission in wireless networks by using the broadcasting method. This enables each subflow data to be delivered by multiple relays. We adapted OR on a number of MPTCP protocols namely, traditional MPTCP, Multipath TCP Traffic Splitting Control (MPTCP-TSC) and Redundant MPTCP (ReMP TCP) in an Internet of Things (IoT) environment. The results show that OR-based MPTCP schemes outperform existing schemes. We further compared the OR-based MPTCP protocols in terms of startup delay and energy efficiency. We found that ReMP TCP is better than other schemes in all scenarios.

Keywords Opportunistic routing (OR) · MPTCP · Internet of things (IoT) · Number of transmissions

1 Introduction

Next-generation networks such as the Internet of Things (IoT), smart healthcare, and intelligent transportation systems generate a massive amount of traffic with stringent Quality of Service (QoS) requirements such as high throughput, low delay and packet-loss [1]. Regular Transmission Control Protocol (TCP) is originally designed for wired networks and adapted in wireless networks which gives rise to some issues such as high packet loss and delay [2]. This is due to the characteristics of wireless channels such as high error rate, interference, fading, obstructions etc [3]. In addition, there are a lot of algorithms have been proposed to enhance TCP performance. For instance, FAST TCP which is designed to consider queuing delay as a congestion signal rather than packet loss probability [4]. Moreover, TCP Fast Open that significantly boost the connection speed of TCP by avoiding the three-way handshake process [5]. Further, Proportional Rate Reduction (PRR) which is a TCP novel method introduced to improve recovery procedure [6]. However, TCP lacks fault tolerance property which is important in next-generation communications [3]. When path failure occurs in the connection, data will be lost, thus, TCP has to reset a new connection. Furthermore, TCP does not support multi-homed terminals such as smartphones, tablets and laptops which are equipped with multiple heterogeneous interfaces available for transmission. This motivate the Internet Engineering Task Force (IEFT) to introduce Multi-Path Transmission Control Protocol (MPTCP) that enables concurrent traffic forwarding on multiple paths using multiple network interfaces (e.g., Wi-Fi, 5G/LTE and Ethernet). MPTCP aims to increase bandwidth and throughput while maintaining a reliable communication between two ends [7]. In fact, MPTCP is originally designed to satisfy three objectives: (1) increase throughput; (2) load balancing and (3) do not harm [8]. Throughput increment can be achieved by bandwidth aggregation; and load balancing is attained by moving the traffic form congested paths while do not harm means that MPTCP

✉ Mohammed Aljubayri
mohammed.aljubayri@kcl.ac.uk

[1] Departemnt of Engineering, King's College London, Strand, London WC2R 2LS, UK

must be fair to other TCP flows. Figure 1 shows the basic scenario of MPTCP communication in transport layer. In addition, MPTCP utilises numerous TCP paths within one connection, each is called a subflow. This offers the opportunity of path diversity that is not available in conventional TCP [9]. Although MPTCP exploits path diversity, it uses a single TCP socket from the perspective of the application, therefore, the application layer in MPTCP remains unmodified. MPTCP is widely deployed and has no issues with middle boxes such Network Address Translations (NATs), firewalls and proxies since it is compatible with regular TCP [10].

MPTCP gained a lot of interests in industry and academia due to its improvements in network throughput. For business purposes, MPTCP is being used in Siri, the digital assistance provided by Apple. It is supported in iPhone Operating System (iOS) version 7.0 and later versions. Moreover, Samsung supports MPTCP in its products such as Galaxy S6 to offer seamless communication and better service [3]. In academia, MPTCP was implemented and evaluated in [11] and demonstrated better performance than conventional TCP. In [12], the authors proposed MPTCP-TSC (Traffic Split Control) as a cost minimization problem subject to end-to-end delay constraint. A greedy heuristic is presented that allocates the transmission rate for every sub-flow. The simulation results showed enhanced performance in comparison with MPTCP. The authors in [13] proposed a scheme known as ReMP TCP to reduce latency. The scheme sends redundant packets over parallel sub-flows to diminish the packet drop probabilities and increase reliability. An experiment was conducted using a simulation as well as a real-world mobile scenario, and it was concluded that the ReMP TCP outperforms regular MPTCP. Nevertheless, the existing MPTCP schemes [11–13] have high number of transmissions which contribute to high delay. Number of transmissions refers to the number of times a packet is transmitted/re-transmitted from source to destination [15]. The reason for the high number of transmissions in the above MPTCP protocols is the use of traditional routing where a lot of re-transmissions can occur particularly in lossy links [14]. To address this challenge, we take advantage of the Opportunistic Routing (OR) technique.

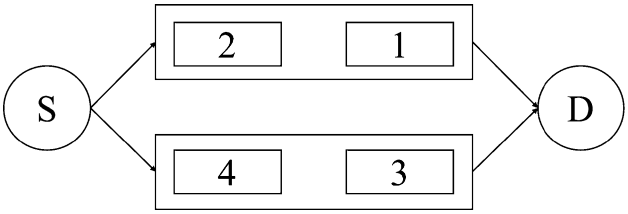


Fig. 1 An illustration of the MPTCP scenario, S is the source and D is the destination

OR is a routing model used to increase the delivery rate and reliability of data transmission in wireless networks [15]. It selects the next routing path at the transmission time by using the broadcasting method. Moreover, in OR, source node creates a list of forwarders (relays) and any adjacent node, which hears the data transmission, can be considered as a forwarder. The source node then can prioritise the forwarders according to certain conditions. In contrast, traditional routing defines the routing path before transmission which can be insufficient particularly in highly dynamic networks. Figure 2 presents the scenario of OR as well as traditional routing in which the source S forwards the traffic to the destination D. The nodes R1, R2 and R3 are acting as relays or forwarders. In traditional forwarding mechanism, the route S-R1-D is selected before sending data. All traffic must use this route until the last packet is successfully delivered to D. In OR, the source S can select R2 or R3 as a forwarder at the transmission time. Therefore, either S-R2-D or S-R3-D route can be utilised.

In this paper, we introduce OR to the protocols [11–13] to reduce the number of transmissions required to successfully deliver a packet to the destination. This will lead to delay reduction and efficient use of network resources in the IoT scenario. In addition, to best of our knowledge, this is the first novel work which combines Multipath TCP with OR. The key contributions of the paper are as follow:

1. We adapted OR to MPTCP, MPTCP-TSC and ReMP TCP in order to reduce the number of transmissions compared to the existing schemes.
2. End-to-end delay is decreased in the existing schemes by exploiting OR technique.
3. We focused on multimedia data transmission and newly proposed schemes reduce the start-up delay by requiring a smaller number of transmissions for the first video frame.
4. Energy efficiency of the protocols is evaluated and our design diminishes the energy costs required for data delivery.

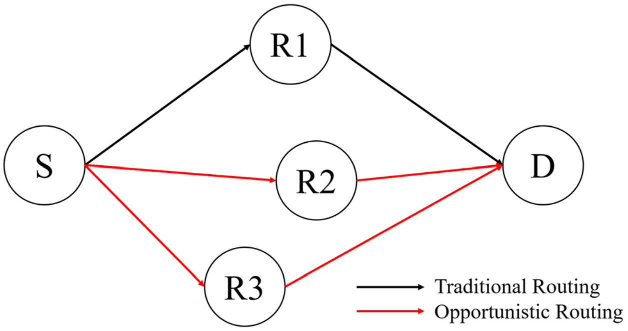


Fig. 2 An illustration of opprtunistic routing and traditional routing procedure

The simulation results illustrate that ReMP TCP has the lowest number of transmissions, end-to-end delay, start-up delay, and energy cost compared to the other protocols. Furthermore, the selected schemes have shown better performance when using the OR technique in the IoT network. The rest of the paper is organized as follows. In Sect. 2, we present related work and a brief analysis on this work. The system model is introduced in Sect. 3. The results of the experiment are presented in Sect. 4. In Sect. 5, we present the conclusion and outline areas for future work.

2 Related work

Wireless communication has experienced exponential growth over last decade. The advancement in next-generation communication networks, such as IoT, have led to a huge increase in the use of smart devices (i.e. smartphones, tablets, and smart-watches). Therefore, there is a need for a promised protocol, such as Multipath TCP, to control the traffic. MPTCP is well studied in the literature and numerous researchers have proposed to develop the protocol. The authors in [16] studied MPTCP and concluded that it has higher throughput compared to regular TCP. In [17], the researchers proposed an MPTCP scheme based on the round-trip time (RTT) policy that forwards packets on the path with the lowest RTT. However, the approach is not efficient in the case of lossy networks. The authors in [18] introduced an MPTCP-based scheme which forwards the packets on the fastest path instead of waiting for the lowest RTT route. The results show that forwarding the packet on the fastest paths improved the performance of the protocol compared to waiting for the lowest RTT paths when the difference between RTTs is large. In addition, the literature in [19] explores the heuristics based on transmission rate, loss rate and delays and concluded that this approach performs better than the lowest RTT approach.

The authors in [20] explored the challenges of deploying MPTCP. Firstly, MPTCP can reduce the throughput of the network due to packet re-ordering issues. Moreover, the number of sub-flows can cause a buffer overflow problem at the receiver side. In [21], MPTCP- Pipeline Network Coding (MPTCP-PNC) was proposed to solve the problem of packet reordering. MPTCP-PNC introduced cheap coding coefficient rules that reduce encoding and decoding delay by progressively coding and decoding packets without waiting for the whole block (group of packets) on both sides, sender and receiver. This means when the transport layer receives data from the upper layer and converts it into blocks or groups, whichever packets come to the encoder are encoded and sent without waiting for the whole group. In addition, packets are assigned to paths according to path quality. On the receiver side, the same process occurs where the packets are decoded and sent to the application layer regardless of the other packets of the same group. Furthermore, the Fountain code-based Multipath TCP (FMTCP) is another way to use coding techniques to recover lost data, improve delay, and increase throughput [22]. Fountain code generates encoded symbols across sub-flows based on path quality. The simulation results illustrate that FMTCP achieves better performance than MPTCP. Furthermore, in [23] an algorithm was proposed, called Stochastic Earliest Delivery Path First (S-EDPF), which combines scheduling and random linear coding using forward error correction (FEC). The authors provided end-to-end performance measurement and the result obtained demonstrates that the delay impact of reordering and packet loss is mitigated. However, coding and decoding model introduces computational complexity to systems.

In [24], the impact of latency in MPTCP in mobile-based applications was analysed. The results show that performance degrades in the case of high traffic generation where modern schedulers are required. Intelligent delay-aware packet scheduling (DAPS) was proposed to send future and current packets at the same time [25]. This means that some sup-flows will forward current packets where others simultaneously forward future packets. As the authors explained, applying this approach achieved a 77% reduction in the receiver's buffer, 63% decrease in the application delay and increased the overall throughput by 42 %. In [26], the authors presented the Blocking Estimation (BLEST) algorithm to minimize Head of Line blocking (HoL) in MPTCP. The BLEST scheduler dynamically adjusts the flows that can cause HoL which leads to better performance according to the simulation results. Nevertheless, in some cases the sup-flows in [25, 26] become idle. The lengthy idle period resets the congestion window (cwnd), hence, underutilises the network resources. This issue was addressed by [27] where the completion time of sub-flows was considered and the problem of underutilisation of flows was solved by reducing the long idle periods. In [28], the researchers introduced a new data transmission model by considering the path symmetry. The model adjusts the path selection for different data flows while predicting the volume of data. This technique achieved short completion time of the MPTCP sub-flows according to the simulation results. In [29], the authors proposed an MPTCP- based weighted round robin scheduler that integrated a load balancing approach and achieved optimal performance; however, the proposed scheme has a fairness issue. The authors in [30] proposed an MPTCP based approach for satisfying the QoS requirement of the network. The traffic is forwarded on the path considering queuing delay, RTT and congestion state.

The results show that higher QoS is achieved compared to other schedulers.

The authors in [31] used MPTCP for resource pooling and a new metric that expects throughput was calculated for each sub-flow. The results showed higher throughput and lower network congestion. In addition, the authors in [32] dynamically adjusted the congestion window according to the requirement of the application. This method results in lower packet loss and better throughput. The authors in [33] designed a framework operating at the receiver-side to optimally allocate a transmission rate for each sup-flow in order to reduce the application-level delay. The receiver sends 3-duplicate acknowledgments (ACKs) to the sender to adjust the congestion window if the current cwnd size is greater than a threshold. However, this process is not practical in IoT networks. In [34], the authors proposed an MPTCP algorithm known as LIA (Linked Increased Algorithm) in which the sub-flows are managed to satisfy two conditions. The first condition is that MPTCP should obtain no more capacity than a single path TCP flow in a shared bottleneck. The second condition is that MPTCP throughput using multiple flows should achieve at least as much throughput as traditional TCP. The LIA scheme does not fully consider the friendliness issue while forwarding the traffic on multiple sub-flows. The authors in [35] proposed an MPTCP congestion control algorithm that adaptively decouples sub-flows which are not sharing the bottleneck to maximize the throughput.

The existing MPTCP performs well for traditional routing-based networks. However, such a routing method is shown to have higher number of transmissions compared to OR which result in delay [36]. In next-generation networks such as IoT, low delay is a significant requirement, thus, traditional routing-based MPTCP schemes cannot fully satisfy the need. Therefore, a research gap exists to improve the protocol. We take the opportunity to adapt OR in MPTCP-based schemes and enhance the IoT wireless sensors network. To best of our knowledge, this is the first novel work which combines Multipath TCP with Opportunistic Routing.

3 System model

The components of the IoT networks are identified in three folds as follows: (1) Hardware—which includes sensors and communication equipment, (2) Middleware—that refer to data storage and analysis (i.e., internet gateways), and (3) Presentation—which means data visualisation (i.e., applications) [37]. Figure 3 illustrates the general IoT network architecture. In this paper, only the network between the sensors and the middleware is studied. We consider a Wireless Sensors Network (WSN) with 10 wireless nodes distributed uniformly as shown in Fig. 4. The sensors are deployed in a network area of 180 × 180 m^2. The S node is the source sensor and D is the destination node. The D node in the proposed model is acting as an IoT gateway. It can be a smartphone that is connected to the Internet using Wi-Fi or 5G cellular communication. S is installed with a camera which senses DHL trucks and identifies them by their plate number, records a video and sends it to D to take an action. The D node will act accordingly by processing the video and sending it to the remote server to inform DHL that the above-identified truck has arrived. Nevertheless, we are only considering the communication between the source sensor and the gateway, thus, the study of the network afterward is outside the scope of this paper. In the IoT network, only the S and the D nodes are MPTCP capable which means they can transmit or receive data through different paths simultaneously. Both the source and the gateway have 2 Wi-Fi interfaces ready for transmission. All other sensors act as relays. The links in the network have packet loss probability of 0.4.

In our work, we assume that there is no interference between the sensor nodes (e.g., using OFDMA communication channels). Also, the D node is assumed to broadcast the acknowledgement to avoid redundant transmissions. For example, if D successfully received a packet with sequence number 1, it would broadcast the acknowledgement of this packet to the neighbors, therefore, they can discard any packet with the same sequence number.

As we mentioned, number of transmissions refers to the number of times a packet is transmitted/re-transmitted from source to destination. Taking this into consideration, we can define end-to-end delay T as

$$T = \sum_{n=1}^{N} NT_n.TD \tag{1}$$

where N is total the number of packets, NT_n is the number of transmissions for the nth packet and TD is the transmission delay. Transmission delay is known as the time taken by a packet to be successfully transferred into the communication link. Thus, TD can be expressed as

$$TD = \frac{PS}{TR} \tag{2}$$

where PS and TR are the packet size and the transmission rate respectively.

In addition, the startup delay considered in this paper is evaluated based on the number of transmissions. We assume that 45 packets are required to display the first video frame (the first picture in a video stream).

Further, energy efficiency in this work is also calculated based on the number of transmissions. We consider the

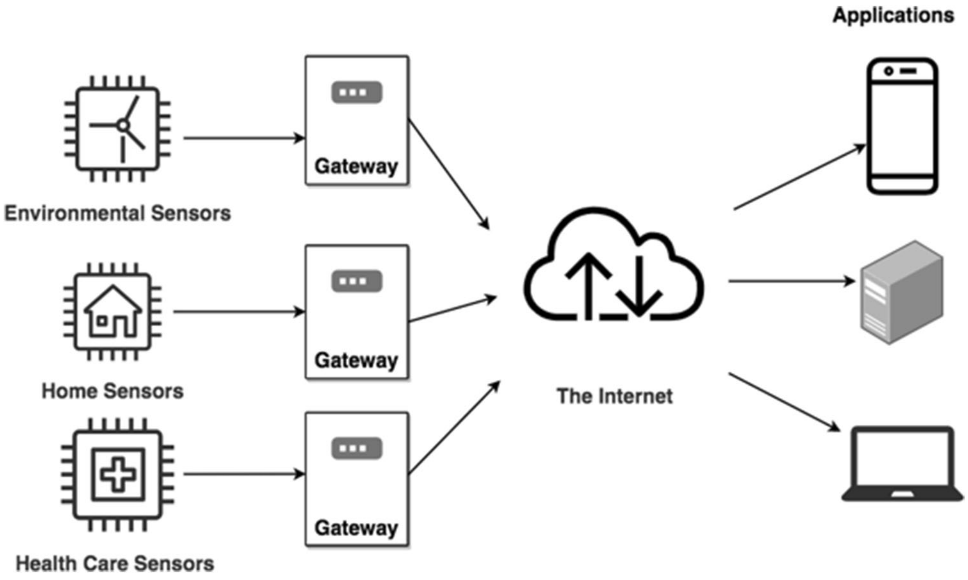


Fig. 3 IoT architecture consistes of hardware (sensors,communiation stack), middleware (gateways) and presentation (applications)

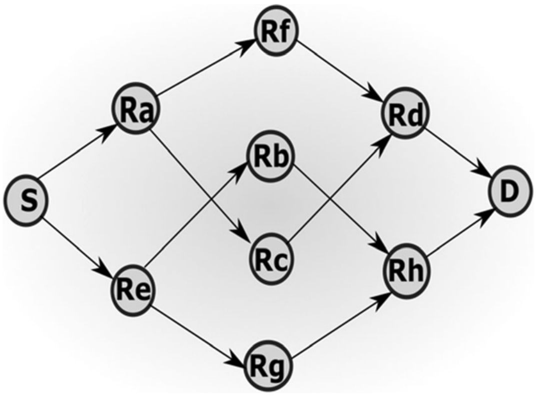


Fig. 4 Proposed OR-based MPTCP IoT network model

total energy consumption as the summation of number of transmissions for each packet multiplied by the transmission power. Hence, total energy consumption E is given as:

$$E = \sum_{n=1}^{N} NT_n.TP \tag{3}$$

where N is the total number of packets, NT_n is the number of transmissions for the *nth* packet and TP is the transmission power.

4 Preliminary experimental results

The simulation tool used for evaluating the performance of the proposed OR-based MPTCP is MATLAB 2020. We compare the selected schemes with respect to number of data transmissions and number of packets. A total of 1000 packets are being transmitted to the destination. In addition, to diminish network overhead, the maximum number of forwarders in the forward list of each node is set to two forwarders. They are selected on random basis to reduce the complexity of the design. The simulation parameters can be seen in Table 1. In the following subsections, we will explain the simulation results.

4.1 Schemes without OR

We simulated MPTCP, MPTCP-TSC, ReMP TCP, as well as regular TCP to obtain precise results. The protocols are simulated without exploiting the Opportunistic Routing technique to analyse the schemes in our IoT network. It can be seen from Fig. 5 that TCP has the worst performance compared to all MPTCP schemes. The reason is that TCP is a single path protocol and this result in higher number of re-transmissions. The MPTCP performance is better compared to the TCP protocol as the single flow splits into

Table 1 Simulation parameters

Simulation parameter	Value
Simulation tool	MATLAB 2020
Coverage area	180×180 m^2
Transmission range	45 meters
Loss probability	0.4
Transmission power (*TP*)	15 dBm
Number of packets (*N*)	1000
Number of paths (in the MPTCP)	2
Communication method	Wi-Fi
Transmission rate (*TR*)	6 Mbps
Packet size (*PS*)	1 KB
Maximum number of forwarders	2

multiple parallel flows and number of transmissions gets lower. The performance of the MPTCP-TSC has shown good results and achieved an acceptable number of transmissions in comparison with TCP and MPTCP. This is because the MPCP-TSC allocates a transmission rate to each sub-flow taking into account the delay constraints. The ReMP TCP has the best performance over all the schemes. The previous Multipath TCP schemes use all available paths for data transmission where ReMP TCP uses some paths for redundant data to reduce re-transmission. Therefore, ReMP TCP is the best scenario.

Furthermore, we evaluated the performance of each protocol with respect to end-to-end delay. Figure 6 shows that TCP achieved the highest delay. MPTCP and MPTCP-TSC protocols perform similarly at the beginning because the number of packets is small which is not enough to illustrate the difference between the two schemes. However, when the number of packets reach about 300, MPTCP-TSC demonstrates lower delay than MPTCP. The reason is that MPTCP-TSC considers path characteristics before transmission, therefore, unnecessary delay can be avoided. In contrast to MPTCP and MPTCP-TSC, ReMP TCP duplicates packets and send them over all available paths simultaneously. Although this approach results in bandwidth underutilisation, it achieved the lowest delay in our experiment.

4.2 Schemes with OR

We improve the protocols by considering OR method. Nevertheless, before integrating OR to the existing schemes, we first simulated OR alone without any transport layer protocol to evaluate its performance. Figure 7 shows the results of our experiment in terms of number of transmissions. We found that OR was the worst scenario in the simulation. This is because OR does not consider flow control between the sender and receiver which lead to frequent packet drops, therefore, frequent re-transmissions. Further, transport layer schemes improved significantly. TCP with OR reduced the transmission count by 12.7% in comparison to the previous result. OR-based MPTCP achieved 12.4 % reduction in number of transmissions compared to plain MPTCP. The MPTCP-TSC scheme using OR decreases total transmissions by 10% in comparison with existing MPTCP-TSC. Lastly, when ReMP TCP was combined with the OR, the number of transmissions dropped by 8.4% compared to ReMP TCP without OR. Additionally, it is shown that ReMP TCP outperforms the above schemes with and without OR. Table 2 shows the

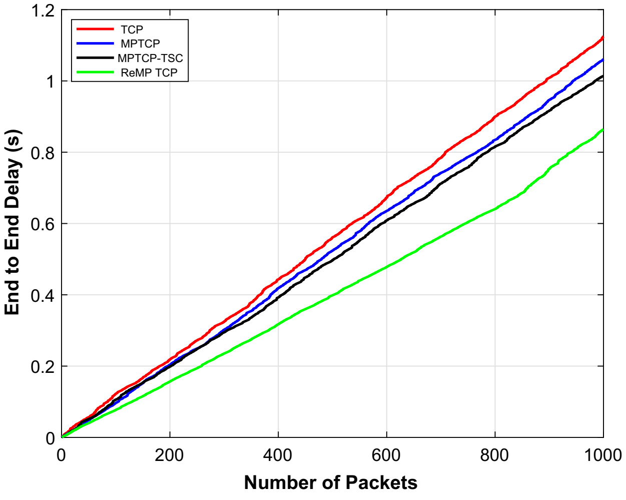


Fig. 6 End-To-End delay evaluation (in seconds) for each protocol with respect to number of packets without exploiting OR technique

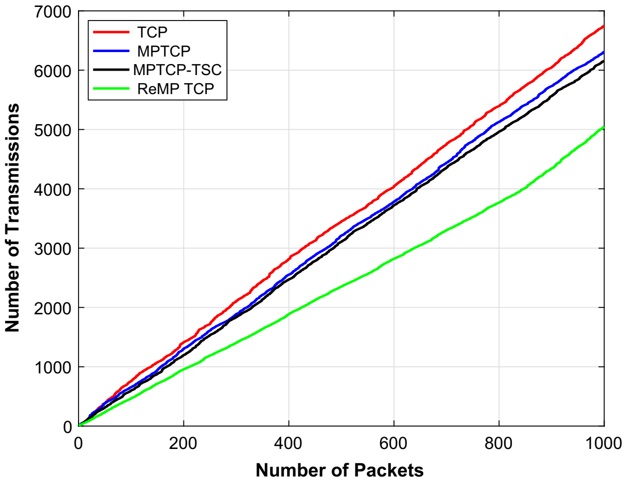


Fig. 5 A comparison of benchmark schemes in number of transmissions versus number of packets generated

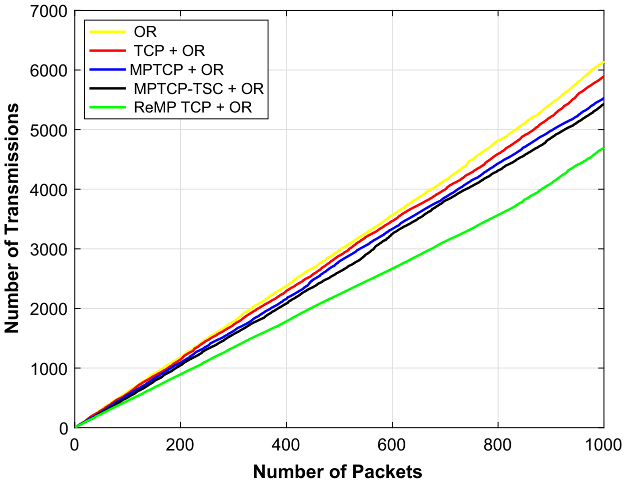


Fig. 7 A comparison of the OR-based MPTCP protocols schemes in number of transmissions versus number of packets generated

average number of transmissions attained by each scheme in the simulation.

Moreover, we evaluated end-to-end delay for OR itself and for each OR-based MPTCP schemes as shown in Fig. 8. It is illustrated that OR was the worst case which is similar to the result in previous sub-section. In this simulation, all schemes required less than 1 s to transmit 1000 packets. In comparison with Fig. 6, ReMP TCP delay dropped from 0.86 to 0.76 s where MPTCP-TSC measurement diminished from 1.01 to 0.90 s. Also, MPTCP delay reduced from 1.06 to 0.92 s when exploiting OR technique and TCP performance improved to require 0.97 s instead of 1.12 in the previous result. Therefore, our proposed schemes are demonstrated to be more effective and sufficient in terms of demanding small number of transmissions and low delay in the IoT network.

We further extend our work to demonstrate the number of transmissions required for each OR-based MPTCP protocol to start-up a video at the destination node. As we mentioned before, we assume that 45 packets are required to display the first video frame. Figure 9 shows the comparison between the schemes and it can be observed that TCP is the worst case. This is due to single path utilization where frequent packet loss occurred. MPTCP performs better than TCP with 28 transmissions reduction. This shows the advantage of splitting a single flow into several sub-flows using network resources. The MPTCP-TSC forwards the same traffic to the destination with a specific transmission rate allocated for each sub-flow. This resulted in requiring 17 transmissions lower than MPTCP. Finally, the ReMP TCP achieved the lowest number of transmissions required to run the first frame with a 15 transmission reduction from MPTCP-TSC. Table 3 illustrates the average number of transmissions for each protocol. Our newly designed schemes showed better performance than the schemes without OR and diminished the start-up delay by requiring a smaller number of transmissions for the first video frame.

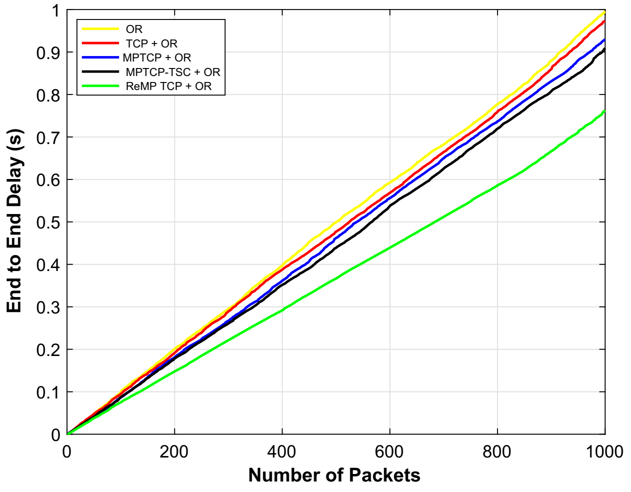


Fig. 8 End-To-End delay evaluation (in seconds) for each OR-based MPTCP scheme with respect to number of packets

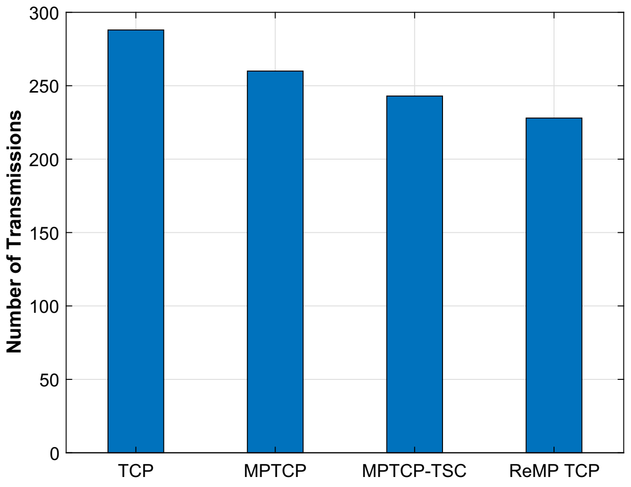


Fig. 9 Demonstration of the number of transmissions required to display the first video frame (45 packets) at the destination for each protocol

The last simulation result obtained is energy efficiency comparison. The transmission power *TP* for each node is 15 dBm. We calculated the energy cost according to equation (3). Figure 10 shows the evaluation of TCP, MPTCP, MPTCP-TSC and ReMP TCP in terms of total energy cost. In our model, TCP has the highest power cost due to the high number of data transmissions using a single transmitter. The MPTCP protocol mitigates the cost by 6.0% in comparison with traditional TCP. The MPTCP-TSC reduces it further and utilizes 2.1% lower energy than MPTCP. The ReMP TCP attained a 15.9% decrease in the total cost compared to MPTCP-TSC. Thus, the performance of ReMP TCP is better than all other schemes.

Table 2 The average transmissions count of selected schemes with and without OR

Schemes	Number of transmissions without OR	Number of transmissions with OR
ReMP TCP	5056	4630
MPTCP-TSC	6100	5430
MPTCP	6332	5546
TCP	6752	5890
OR	–	6090

Table 3 The average transmissions count of OR-based MPTCP protocols to successfully send the first video frame to the destination

Schemes	Number of transmissions
ReMP TCP	228
MPTCP-TSC	243
MPTCP	260
TCP	288

Table 4 The total energy consumption of OR-based MPTCP protocols in dBm

Schemes	Total energy costs in dBm
ReMP TCP	69450
MPTCP-TSC	81450
MPTCP	83190
TCP	88350

Table 4 illustrates the total energy costs for each protocol. In this simulation, we illustrated that OR-based MPTCP protocols provide efficient use of network resources and are convenient for IoT applications.

To summarize the simulation results, it can be observed that OR technique provides considerable enhancement to the existing protocols and reduces the number of transmission which consequently reduces delay. In addition, it is energy efficient and provides better utilisation of IoT network resources. Thus, OR with MPTCP protocols is a promising method for the next generation networks.

5 Conclusion and future work

The Internet of things (IoT) aims to connect billions of smart devices that can operate independently without human interaction. The existing MPTCP protocols perform well for traditional routing-based networks. However, such a routing method is shown to have high number of transmissions, compared to Opportunistic Routing, which result in delay. In next-generation networks such as IoT, low delay is a significant requirement, thus, traditional routing-based MPTCP schemes cannot fully satisfy the need. Therefore, a research gap exists to improve the protocol. We adapted OR in the existing MPTCP, MPTCP-TSC and ReMP TCP to address this challenge. The OR is a networking model in which the traffic can be forwarded at the transmission time to any relay node that hear the transmission, therefore, increasing the reliability of data transmission in a network. The simulation results show that the OR-MPTCP schemes reduced the number of transmissions compared to the existing schemes without using the OR method. Furthermore, the comparison of the MPTCP, MPTCP-TSC and ReMP TCP shows that ReMP TCP has the lowest number of transmissions, the lowest delay and is energy efficient compared to other schemes. Thus, our design can be an effective approach in improving the performance of the IoT networks. In future work, we could utilise machine learning techniques, such as Deep Reinforcement learning technique, that can further enhance OR. Moreover, the proposed OR-based MPTCP schemes are simulated on small network topology and its scalability in case of large IoT networks need further investigations. Thus, this can be considered as a future research.

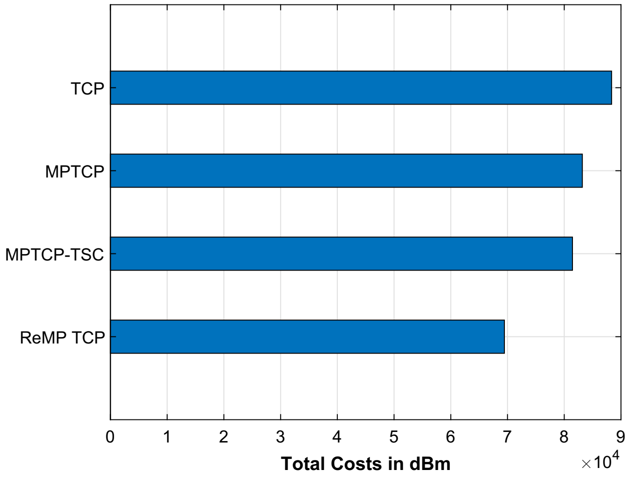


Fig. 10 Total energy consumption of OR- based MPTCP schemes

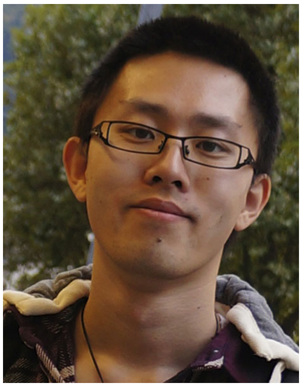
References

1. Verma, S., Kawamoto, Y., Fadlullah, Z. M., Nishiyama, H., & Kato, N. (2017). A survey on network methodologies for real-time analytics of massive IoT data and open research issues. *IEEE Communications Surveys & Tutorials, 19*(3), 1457–1477.
2. Tian, Y., Xu, K., & Ansari, N. (2005). Tcp in wireless environments: Problems and solutions. *IEEE Communications Magazine, 43*(3), S27–S32.

3. Sharma, V. K., Verma, L. P., & Kumar, M. (2019). Cl-adsp: Cross-layer adaptive data scheduling policy in mobile ad-hoc networks. *Future Generation Computer Systems, 97,* 530–563.
4. Wei, D. X., Jin, C., Low, S. H., & Hegde, S. (2006). Fast tcp: Motivation, architecture, algorithms, performance. *IEEE/ACM Transactions on Networking, 14*(6), 1246–1259.
5. Sy, E., Mueller, T., Burkert, C., Federrath, H., & Fischer, M. (2020). Enhanced performance and privacy for tls over tcp fast open. *Proceedings on Privacy Enhancing Technologies, 2020*(2), 271–287.
6. Mittal, V., Jain, V., & Tahiliani, M. P. (2018). Proportional rate reduction for ns-3 tcp. In *Proceedings of the 10th workshop on ns-3* (pp. 9–15).
7. Ford, A., Raiciu, C., Handley, M., & Bonaventure, O. (2013). Tcp extensions for multipath operation with multiple addresses. Technical report, RFC 6824.
8. Usach, R. G. and Kühlewind, M. (2012). Implementation and evaluation of coupled congestion control for multipath tcp. In *Meeting of the European network of universities and companies in information and communication engineering* (pp. 173–182). Springer.
9. Paasch, C., Ferlin, S., Alay, O., & Bonaventure, O. (2014). Experimental evaluation of multipath tcp schedulers. In *Proceedings of the 2014 ACM SIGCOMM workshop on Capacity sharing workshop* (pp. 27–32).
10. Bonaventure, O., & Seo, S. (2016). Multipath tcp deployments. *IETF Journal, 12*(2), 24–27.
11. Wischik, D., Raiciu, C., Greenhalgh, A., & Handley, M. (2011) Design, implementation and evaluation of congestion control for multipath tcp. In *NSDI* (Vol. 11, pp. 8).
12. Park, S.-Y., Joo, C., Park, Y., & Bank, S. (2014). Impact of traffic splitting on the delay performance of mptcp. In *2014 IEEE international conference on communications (ICC)* (pp. 1204–1209). IEEE.
13. Frommgen, A., Erbshäußer, T., Buchmann, A., Zimmermann, T., & Wehrle, K. (2016). Remp tcp: Low latency multipath tcp. In *2016 IEEE international conference on communications (ICC)* (pp. 1–7). IEEE.
14. Lu, M.-H., Steenkiste, P., & Chen, T. (2007). Video transmission over wireless multihop networks using opportunistic routing. In *Packet video 2007* (pp. 52–61). IEEE.
15. Menon, V. G. (2019). A comprehensive survey on opportunistic routing protocols for manets: Issues, challenges and future directions. Preprints, 2019070239. https://doi.org/10.20944/preprints201907.0239.v1
16. Chen, Y.-C., Lim, Y.-S., Gibbens, R. J., Nahum, E. M., Khalili, R., & Towsley, D. (2013). A measurement-based study of multipath tcp performance over wireless networks. In *Proceedings of the 2013 conference on internet measurement conference* (pp. 455–468).
17. Ni, D., Xue, K., Hong, P., & Shen, S. (2014). Fine-grained forward prediction based dynamic packet scheduling mechanism for multipath tcp in lossy networks. In *2014 23rd international conference on computer communication and networks (ICCCN)* (pp. 1–7). IEEE.
18. Hwang, J., & Yoo, J. (2015). Packet scheduling for multipath tcp. In *2015 seventh international conference on ubiquitous and future networks* (pp. 177–179). IEEE.
19. Kimura, B. Y., Lima, D. C., & Loureiro, A. A. (2017). Alternative scheduling decisions for multipath tcp. *IEEE Communications Letters, 21*(11), 2412–2415.
20. Raiciu, C., Paasch, C., Barre, S., Ford, A., Honda, M., Duchene, F., Bonaventure, O., & Handley, M. (2012). How hard can it be? Designing and implementing a deployable multipath {TCP}. in 9th {USENIX} symposium on networked systems design and implementation ({NSDI} 12) (pp. 399–412).
21. Xu, C., Wang, P., Xiong, C., Wei, X., & Muntean, G.-M. (2016). Pipeline network coding-based multipath data transfer in heterogeneous wireless networks. *IEEE Transactions on Broadcasting, 63*(2), 376–390.
22. Cui, Y., Wang, L., Wang, X., Wang, H., & Wang, Y. (2014). Fmtcp: A fountain code-based multipath transmission control protocol. *IEEE/ACM Transactions on Networking, 23*(2), 465–478.
23. Garcia-Saavedra, A., Karzand, M., & Leith, D. J. (2017). Low delay random linear coding and scheduling over multiple interfaces. *IEEE Transactions on Mobile Computing, 16*(11), 3100–3114.
24. Grinnemo, K.-J., & Brunstrom, A. (2015). A first study on using mptcp to reduce latency for cloud based mobile applications. In *2015 IEEE symposium on computers and communication (ISCC)* (pp. 64–69). IEEE.
25. Kuhn, N., Lochin, E., Mifdaoui, A., Sarwar, G., Mehani, O., & Boreli, R. (2014). Daps: Intelligent delay-aware packet scheduling for multipath transport. In *2014 IEEE international conference on communications (ICC)* (pp. 1222–1227). IEEE.
26. Ferlin, S., Alay, Ö., Mehani, O., & Boreli, R. (2016). Blest: Blocking estimation-based mptcp scheduler for heterogeneous networks. In *2016 IFIP networking conference (IFIP networking) and workshops*. (pp. 431–439). IEEE.
27. Lim, Y.-S. , Nahum, E. M. , Towsley, D., & Gibbens, R. J. (2017). Ecf: An mptcp path scheduler to manage heterogeneous paths. In *Proceedings of the 13th international conference on emerging networking experiments and technologies* (pp. 147–159).
28. Dong, P., Yang, W., Tang, W., Huang, J., Wang, H., Pan, Y., & Wang, J. (2018). Reducing transport latency for short flows with multipath tcp. *Journal of Network and Computer Applications, 108,* 20–36.
29. Choi, K. W., Cho, Y. S., Lee, J. W., Cho, S. M., & Choi, J. (2017). Optimal load balancing scheduler for mptcp-based bandwidth aggregation in heterogeneous wireless environments. *Computer Communications, 112,* 116–130.
30. Noda, K., & Ito, Y. (2018). Proposal of novel mptcp congestion control to suppress qos fluctuation for webqoe improvement. In *2018 IEEE 8th international conference on consumer electronics-Berlin (ICCE-Berlin)* (pp. 1–3). IEEE.
31. Liu, Y., Qin, X., Zhu, T., Chen, X., & Wei, G. (2019). Improve mptcp with sdn: From the perspective of resource pooling. *Journal of Network and Computer Applications, 141,* 73–85.
32. Lubna, T., Mahmud, I., & Cho, Y.-Z. (2020). D-lia: Dynamic congestion control algorithm for mptcp. *ICT Express, 6,* 263–268.
33. Park, S.-Y., Joo, C., Park, Y., & Bahk, S. (2015). Minimizing application-level delay of multi-path tcp in wireless networks: A receiver-centric approach. In *2015 IEEE 23rd international conference on network protocols (ICNP)* (pp. 245–255). IEEE.
34. Raiciu, C., Handley, M., & Wischik, D. (2011). Coupled congestion control for multipath transport protocols. Technical report, IETF RFC 6356.
35. Ferlin, S., Alay, Ö., Dreibholz, T., Hayes, D. A., & Welzl, M. (2016). Revisiting congestion control for multipath tcp with shared bottleneck detection. In *IEEE INFOCOM 2016-The 35th annual IEEE international conference on computer communications* (pp. 1–9). IEEE.
36. Biswas, S., & Morris, R. (2005). Exor: Opportunistic multi-hop routing for wireless networks. In *Proceedings of the 2005 conference on Applications, technologies, architectures, and protocols for computer communications* (pp. 133–144).
37. Gubbi, J., Buyya, R., Marusic, S., & Palaniswami, M. (2013). Internet of things (IotT: A vision, architectural elements, and future directions. *Future Generation Computer Systems, 29*(7), 1645–1660.

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Mohammed Aljubayri received the B.Sc. degree in Computer Science from King Khalid University, Bisha, Saudi Arabia, in 2007 and the M.Sc. degree in Information Technology from the University of Newcastle, Australia, in 2011. He is working as a lecturer in the College of Computer Science and Information Systems in Najran University, Najran, Saudi Arabia since 2012. He is currently working towards the Ph.D. degree in Telecommunications with the Department of Engineering, Centre for Telecommunications Research, in King's College London, London, U.K. His current research interests include multipath transmission control protocol, network routing, the Internet of Things and wireless communication networks.



Tong Peng received the B.Eng. degree in Electronics Engineering from Liaocheng University, Shandong, China, in 2009, the M.Sc. and Ph.D. degree in Communications Engineering in Electronic Engineering from the University of York, U.K., in 2010 and 2014, respectively. He worked at the Centre for Studies in Telecommunications (CETUC) in the Pontifical Catholic University of Rio de Janeiro (PUC-Rio) as a Research Associate from 2014–2015, and then worked for "Network coded modulation for next generation wireless access networks" Project in the Communications Group, the Department of Electronic Engineering, University of York from 2015–2017. He joined the Centre for Telecommunications Research (CTR), Department of Informatics, King's College London, in 2017 and work as a research associate on "Internet of Silicon Retina" Project. He is now with the Department of Mechanical Electrical and Manufacturing Engineering in Loughborough University, UK, as well as Department of Informatics and Engineering in Zhejiang Ocean University, China. His research interests include space-time coding, network coding, resource allocation and multimedia communications in 5G IoT.

Mohammad Shikh-Bahaei received the B.Sc. degree from the University of Tehran, Tehran, Iran, in 1992, the M.Sc. degree from the Sharif University of Technology, Tehran, in 1994, and the Ph.D. degree from Kings College London, U.K., in 2000. He has worked for two start-up companies, and National Semiconductor Corporation, Santa Clara, CA, USA (currently part of Texas Instruments, Inc.), on the design of third-generation mobile handsets, for which he has been awarded three U.S. patents as an inventor and a co-inventor, respectively. He joined Kings College London as a Lecturer in 2002, where he is currently a Full Professor. He has been engaged in research in the area of wireless communications and signal processing for 25 years both in academic and industrial organisations. His research interests include full-duplex and cognitive dense networking, visual data communications over the Internet of Things, applications in healthcare, and secure communication over autonomous vehicle/drone networks. He is a Founder and the Chair of the Wireless Advanced (formerly, SPWC) International Conference from 2003 to 2018. He is a fellow of IET.