Intelligent TCP Congestion Control Scheme in Internet of Deep Space Things Communication

Arooj Masood, Taeyun Ha, Demeke Shumeye Lakew, Nhu-Ngoc Dao, and Sungrae Cho

***Abstract*—In deep space missions, large-scale deep space information and continuous sensory data are generated, processed, and exchanged over Internet of deep space things (IoDST). Ensuring reliable communication in IoDST requires resolution of challenges engendered by the unique characteristics of IoDST, including long propagation delays, link outages, high error rates, and asymmetric bandwidths. Thus, transmission control protocol (TCP) layer functionalities are crucial for ensuring reliable IoDST communications. However, existing TCP congestion control (CC) protocols present poor performance in IoDST communications, owing to the dependence of traditional window-based CC approaches on pre-configured rules to adjust transmission rate against the aforementioned unique characteristics. In this paper, we propose an intelligent CC scheme called optimistic actor–critic-based TCP congestion control (OAC-TCPCC) to solve the problems of the challenging link conditions in IoDST. OAC-TCPCC dynamically determines optimal congestion window for data transmission over IoDST communication links to maximize the TCP throughput performance and minimize file transfer time. Simulation results reveal that OAC-TCPCC improves the TCP throughput performance by up to 27%, 79%, 92%, 477%, 643%, and 766%; and reduces file transfer time by up to 16%, 37%, 50%, 48%, 58%, and 79%, compared to DRL-CC, TP-Planet, FAST TCP, TCP Peach, TCP BBR, and TCP CUBIC, respectively.**

***Index Terms*—Internet of Deep Space Things, transmission control protocol, congestion control, deep reinforcement learning, optimistic actor–critic**

I. Introduction

OVER the past two decades, advances in space technologies have enabled the realization of deep space exploration missions, such as the Mars exploration mission. These missions produce large volumes of continuous scientific data related to the outer space. The reliable transmission of the big data of such origin over deep space links between planets, spacecrafts, and crewed vehicles requires advanced communication technologies [1]–[3]. Based on future expansion, the Internet of deep space things (IoDST) is envisioned to provide reliable communication services to future deep space exploration missions. Fig. 1 illustrates an IoDST system comprising three networks: interplanetary backhaul, planetary inter-spacecraft network, and planetary network [4], [5]. In particular, the interplanetary backhaul network comprises communication links (direct or multihop links) between the Earth and outer-space planets. The planetary inter-spacecraft network consists of communication links between spacecrafts orbiting around the planets and providing relay services between the Earth and outer-space planets. Finally, the planetary network provides communication links between planetary surface entities, such as landing vehicles and rovers. Among the three aforementioned architectural components, interplanetary backhaul network is characterized by extremely long and variable propagation delays (e.g., ranging between 8.5 min and 40 min based on the orbital positions of the planets with respect to the Mars-to-Earth communication network) [6]. It is also associated with high link error rates, link outages due to orbital obscuration with the loss of line-of-sight, and bandwidth asymmetry in the forward and reverse channels. This network imposes the most challenging problems in achieving reliable data transmission.

Transmission control protocol (TCP) layer functionalities play important roles in ensuring reliable data transmission across an end-to-end connection [2], [3]. To this end, some congestion control (CC) mechanisms were designed as a fundamental principle of TCP that maintain a congestion window (CW) to control the data sending rate over deep space links [5], [7]–[10]. Starting with Akan *et al.* a rate-based additive-increase multiplicative-decrease (AIMD) CC was designed to improve throughput for planetary networks [5]. Following that, Grieco *et al.* proposed an adaptive rate-based CC to improve the goodput over planetary paths [7], [8]. Later, Psaras *et al.* proposed a deep-space transport protocol (DS-TP) that employs double automatic re-transmission (DAR) technique [9] for redundant data transmission. Afterwards, Papastergiou *et al.* proposed a delay- tolerant transport protocol (DTTP), which uses the features of both a delay-tolerant network and the DS-TP technique to improve reliability of data transfer [10]. However, the CC approaches in these studies [5], [7]–[10] were based on pre-configured rules to adjust the transmission rate and are far from optimal i.e., they have unresolved issues such as insufficient usage of available bandwidth, inadaptable to dynamic network situations, and compromising on one or more performance metrics.

It can be concluded from the aforementioned analysis that the TCP performance can be improved if the TCP can learn from past experiences based on the observed values of congestion-related metrics, such as throughput and round-trip time (RTT), during previous interactions with link character-

This research was supported in part by the Chung-Ang University Research Scholarship Grants in 2020, and in part by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT)(No. 2022R1A4A5034130) *(Corresponding authors: Nhu-Ngoc Dao; Sungrae Cho).*

Arooj Masood, Taeyun Ha, Demeke Shumeye Lakew, and Sungrae Cho are with the School of Computer Science and Engineering, Chung-Ang University, Seoul 156-756, Republic of Korea (email: arooj@uclab.re.kr, tyha@uclab.re.kr, demeke@uclab.re.kr, srcho@cau.ac.kr).

Nhu-Ngoc Dao is with the Department of Computer Science and Engineering, Sejong University, Seoul 05006, Republic of Korea (email: nndao@sejong.ac.kr).

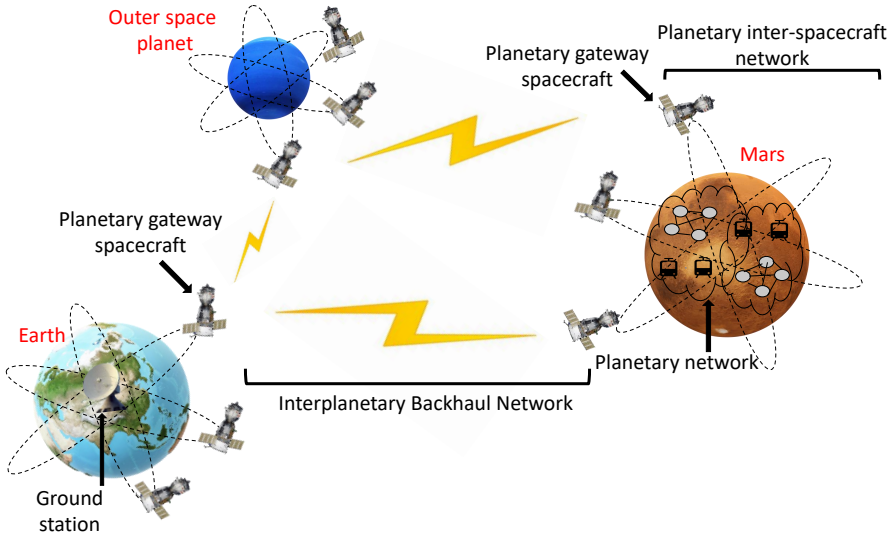


Fig. 1: Architecture of Internet of deep space things (IoDST) system.

istics [11]. There exist learning-based solutions using reinforcement learning (RL) [12], [13] and deep RL (DRL) [14]–[20] for improving the TCP performance in wireless terrestrial networks. However, these CC approaches for wireless terrestrial networks cannot be applied to the interplanetary backhaul network owing to the abovementioned distinctive features of its links in IoDST systems. Furthermore, studies based on a deep Q-network (DQN) [14], [15] are constrained to only provide discrete control, and continuous control on the problem cannot be trivially extended. In contrast, the existing actor–critic based-DRL methods in continuous control tasks [16]–[20] are limited by their poor sample efficiency, which makes the deployment of these algorithms costly in systems for which obtaining samples is expensive [21], particularly in IoDST systems in which the knowledge of the age of information is essential. [21] addressed the inefficiencies of existing actor–critic methods and introduced an optimistic actor–critic-based DRL (OAC-DRL) method, which improves samples efficiency of the policy gradient methods by applying the principle of optimism [22]. OAC-DRL samples actions using exploration policy which increases chances of executing more informative actions and trades off between maximizing an upper confidence bound to the critic and constraining the maximum Kullback–Leibler (KL) divergence between the exploration policy and the target policy, which ensures the stability of updates.

In this paper, we focus on the CC problem in IoDST communication and utilize an OAC-DRL method to design an intelligent CC scheme called OAC-TCPCC. OAC-TCPCC determines optimal CW policy for data transmission in interplanetary backhaul networks in IoDST communication to maximize the TCP throughput performance and minimize delay. Our simulation experiments in Section VI show that the proposed OAC-TCPCC scheme outperforms existing window-based CC schemes in terms of throughput, file transfer time, and fairness. Summarizing the primary contributions of this study are as follows:

- First, we formulate the TCPCC problem for the interplanetary backhaul network links in an IoDST communication as a RL problem.
- Second, we establish an actor–critic RL algorithm for TCPCC in IoDST communication to optimize the TCP throughput performance. In particular, we propose an OAC-TCPCC algorithm to search for the optimal policy function for the formulated RL problem comprising continuous state and action spaces. This is because of its distinctive capability of providing efficient exploration in challenging environments, such as IoDST communication, where obtaining samples for evaluating link conditions is critical.
- Third, it is verified that the proposed OAC-TCPCC scheme improves the TCP performance by automatically identifying the optimal CW policy to achieve high throughput performance and reduce delays and segment loss rates.
- Fourth, a simulation is conducted using NS-3 and PyTorch to ensure the validity and accuracy of the results. Compared to DRL-CC [17], TP-Planet [5], FAST TCP [23], TCP Peach [24], TCP BBR [25], and TCP CUBIC [26], the proposed algorithm increases throughput performance by up to approximately 27%, 79%, 92%, 477%, 643%, and 766%, respectively, and reduces file transfer time by up to approximately 16%, 37%, 50%, 48% and 58%, and 79%, respectively.

To the best of our knowledge, this study is the first effort to exploit the optimistic actor–critic-based advanced deep reinforcement learning (OAC-DRL) algorithm to maximize the TCP throughput performance and minimize file transfer time in an interplanetary backhaul network in IoDST communication.

The remainder of this paper is organized as follows. In Section II, we present an overview of the related studies. The problem statement and the RL-based problem formulation are described in Sections III and IV, respectively. Section V describes the fundamental concepts of DRL and presents our OAC-TCPCC algorithm for the IoDST communication. The

details of the implementation and performance evaluation are presented in Section VI. Finally, the paper is concluded and future research directions are given in Section VII.

II. Related Work

CC, a fundamental problem in TCP, has been extensively studied in the context of both wired [27] and wireless networks [28]. However, these approaches cannot be directly applied to interplanetary backhaul links owing to the extremely high propagation delay, random link errors, and the other above-mentioned characteristics of the problem [6], [29]. Over the years, many transport protocols have also been developed for high-speed links [26] and satellite links, which are also characterized by high bandwidth-delay products (BDP) and random link errors [23]–[25]. For instance, Ha *et al.* adopted a cubic function to adjust the CW by searching for spare bandwidth aggressively to achieve high throughput and fairness. Akyildiz *et al.* proposed TCP-Peach [24], which is based on the use of dummy segments to improve the goodput performance and fairness in satellite networks. Wei *et al.* proposed FAST TCP [23], that addressed CC at large window sizes for high-speed long-latency networks. Cardwell *et al.* developed a novel bottleneck bandwidth round-trip propagation time (BBR) CC [25] that periodically estimates the available bandwidth and minimum RTT on the link and regulates CW size to approach an associated optimal operating point (i.e., Kleinrocks' optimal point [30]). However, the CC approaches in these studies [23]–[26] are limited to be applied to high BDP links in which the RTTs are much shorter and the bit error rates are much lower than interplanetary backhaul networks in the IoDST. In addition, frequent link outages and asymmetrical bandwidth problems in interplanetary backhaul links are so complex that achieving an effective control on CW adjustment is unrealistic; hence, they cannot be applied to the interplanetary backhaul links in the IoDST.

A. TCP Congestion Control Methods for IoDST

To overcome the fundamental limitations of TCP over planetary paths, Akan *et al.* proposed a rate-based AIMD CC mechanism, whose parameters were tuned by considering the connection RTT [5]. It exploits low- (N_{low}) and high-priority (N_{high}) NIX segments to obtain congestion decision support by comparing the acknowledgements (ACKs) of these segments. The CC is determined by considering the ratio, $\phi = N_{low}/N_{high}$. When ϕ is less than a decrease threshold (ϕ_d), the CW decreases multiplicatively. When $\phi_d \leq \phi \leq \phi_i$, where ϕ_i is the increase threshold, the CW remains unchanged, and when $\phi \geq \phi_i$ the CW increases additively. However, the transmission of low-priority NIX segments leads to communication overhead on the interplanetary links. Additionally, low-priority segments discard capability is required at the intermediate routers [5]. Grieco *et al.* proposed an adaptive rate control algorithm to improve the utilization of planetary links as a rate-based version of classic TCP CC. Specifically, the algorithm uses a probing phase aimed at utilizing the network available bandwidth and a shrinking phase that helps reduce the input rate in the presence of congestion [7], [8]. However, these approaches [7], [8] do not consider that the received information about the link conditions is old owing to the long RTTs of the connection and the CC based on such past information may not be an appropriate action.

To address the random packet losses due to high bit error rates and the intermittent connectivity due to link outages, Psaras *et al.* proposed a deep-space transport protocol (DS-TP) employing the double automatic re-transmission technique, which sends each segment twice, importing some delay between the original transmission and the re-transmission. It helps to recover lost segments with the re-transmission of original segment. However, transmitting each segment twice is inefficient for bandwidth utilization [9]. In contrast, a delay-tolerant transport protocol (DTTP), which uses the features of both a delay-tolerant network and the DS-TP technique, adds redundancy to time-sensitive segments and, hence, enhances the reliability of data transfer at the cost of the bandwidth [10]. Another contribution to CC schemes suited for delay-tolerant networks was made by Bureleigh *et al.*, who used the bundling approach and a custody-based store-and-forward mechanism [31], [32]. Even though this approach achieves reliable transport over intermittent links, a specifically tailored transport protocol is required for high-performance bundle transport for the interplanetary backhaul network.

The CC approaches in these studies [5], [7]–[10], [31], [32] were based on pre-configured rules to adjust the transmission rate in challenging and complex IoDST communication scenarios; hence, they failed to characterize the network dynamics and throughput performance.

B. DRL-based TCP Congestion Control Methods

In contrast, learning-based methods such as RL and DRL schemes were recently introduced to solve CC problem in wireless terrestrial networks [12]–[20]. For instance, Winstein *et al.* proposed TCP Remy [12], an offline training mechanism to find optimal mappings from observed network conditions to predefined CC rules. However, TCP Remy works well at the cost of prior assumptions about the network and traffic models. Li *et al.* proposed a Q-learning framework called QTCP [13], that enables senders to derive CC policies in an online manner. Xiao *et al.* proposed TCP-Drinc [14], which utilizes one DQN agent for each TCP flow to control its CW. However, when there are multiple TCP flows, TCP-Drinc requires high computing resources. Cui *et al.* proposed a DQN-based CC scheme to predict the BDP of link for the periodic network fluctuation in the high-speed railway (HSR) scenario [15]. On the other hand, Xu *et al.* proposed a cross-layer knowledge (i.e., RSRP), aided CC scheme in the HSR scenario [16]. The proposed scheme is based on deep recurrent deterministic policy gradient (DRDPG) to distinguish between the negative effects caused by congestion and those caused by handover. Xu *et al.* proposed DRL-CC on multi-path TCP (MPTCP) [17]. However, the reward applied in DRL-CC only considers throughput, which might be harmful in some scenarios with low bottleneck bandwidth. Chen *et al.* introduced DRL for CC with radio access network (RAN) information and reward redistribution called DRL-3R [18] that

predicts the current RAN information to avoid RAN-induced congestion and incorporates reward redistribution to improve network performance. Li *et al.* proposed TCP NeuRal adaptive with online changepoint detection (NeuRoc) that adopts an online changepoint detection method to monitor the network situation and uses DRL to update the CC decision. In the beginning, TCP-NeuRoc performs like bottleneck bandwidth round-trip propagation time (BBR [25]), and then moves to the well-trained policy network [19]. Abbasloo *et al.* proposed a plug-in-based approach called DeepCC [20], that stands on top on throughput-oriented TCP schemes to improve the delay performance of it, and learns to trade a bit of throughput to control delay and achieve application's desired delay performance in cellular networks.

However, the CC approaches in these studies [12]–[20] cannot be applied to the interplanetary backhaul networks in an IoDST communication due to the following reasons. First, these approaches were designed for the wireless terrestrial networks and could not capture the distinctive features of the interplanetary backhaul network in an IoDST communication. Second, the CW update is ineffective for adapting to interplanetary backhaul networks in IoDST communication, which may demand a more severe CW increase. However, the existing DRL-based CC, proposed for the wireless terrestrial networks, may require a conservative CW increase. Third, existing actor–critic based-DRL methods for TCP CC are limited by their inefficient exploration, which makes the deployment of these CC approaches costly in IoDST communication. In contrast to these prior works [12]–[20], we consider unique characteristics of the interplanetary backhaul networks in IoDST communication and employ OAC-DRL method, which achieves efficient exploration in challenging environments, to design an intelligent CC scheme for data transmission in IoDST communication.

III. Problem Statement

A. Analysis of Scenario

Fig. 2 illustrates the network scenario considered for the TCPCC problem in the interplanetary backhaul network in IoDST communication. At the core of the mission spacecrafts for other planets and the ground stations at the Earth, planetary gateway spacecrafts and relay spacecrafts are the networking entities providing seamless communication services across the solar system. The IoDST communication requires the reliable transmission of critical mission data over the interplanetary backhaul links. However, these links characterized by extremely long and variable propagation delays, high bit error rates, blackout conditions, and bandwidth asymmetry impose the most challenging problems in achieving reliable data transmission. We consider an interplanetary bottleneck link with a capacity of 1 Mb/s and bit error rates on the order of 10^{-1} with a propagation delay of 8.5 to 40 min for the Mars-Earth communication network [5]. The asymmetries in the bandwidth capacities of the forward and reverse channels are on the order of 1000:1 [6], where the forward channel capacity corresponds to the data rate of the source whereas the backward channel capacity corresponds to the data rate

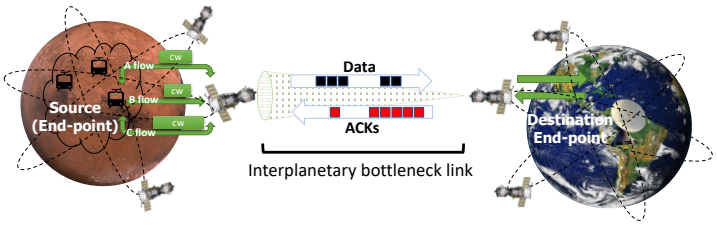


Fig. 2: Network scenario of TCPCC in IoDST.

of the destination. An end host may simultaneously execute more than one TCP connection for sharing an interplanetary bottleneck link. First, the TCP source establishes a connection with its destination, initiates data transmission, and receives feedback from the destination in the form of ACKs, as shown in Fig. 2. Subsequently, the TCP source adjusts its transmission rate based on this feedback, which is determined by the CC algorithm.

However, the TCP source does not know the link conditions or have sufficient information about the congestion in the bottleneck link before determining the CW. The existing TCP CC approaches for interplanetary links initiate data transmission using low data rates and adjust the transmission rate based on static and predefined rules [5], [7]–[10]. However, these approaches are slow to converge to the optimal transmission rate and do not address the short TCP flows in interplanetary backhaul links. For instance, spacecraft navigation services or commands to in-situ nodes mostly feature short TCP flows and may complete data transmission before CC algorithm can have any effect. Thus, short flows suffer from unnecessarily high number of RTTs due to the low throughput achieved. For long TCP flows, adjusting the initial CW does not reduce the RTTs sufficiently to considerably impact flows consisting of hundreds or thousands of round trips. In addition, owing to the extremely long propagation delay, a delayed feedback about the link conditions is received by the TCP source. Thus, a CC decision based on such delayed feedback may lead to inappropriate actions on links with an extremely long propagation delay. Moreover, the packet losses caused by the blackout conditions due to the mobility of the planetary bodies may also mislead the CC decision and affect the corresponding performance of the CC mechanism. Therefore, increasing the transmission rate and decreasing the number of round trips are challenging tasks for both short and long TCP flows, as they require an intelligent CC scheme to address the challenging problems posed by the unique characteristics of the interplanetary backhaul network in IoDST communication.

B. Feasible Approach

To improve the throughput performance on interplanetary backhaul network links by addressing the challenges due to the extremely long and variable propagation delays, high bit error rates, and blackout conditions, we propose to use a learning-based solution using a DRL algorithm for TCPCC. This method is chosen owing to its capability to learn the aforementioned challenges from past interactions with the environment and make decisions without any prior knowledge

of the link conditions. In particular, we utilize an OAC-based advanced DRL algorithm owing to its distinctive capability of providing sample efficiency in challenging environments [21], such as in the IoDST, where collecting data samples to evaluate the link conditions is critical. The OAC-based DRL agent deployed at the source for CC on the forward channel optimizes the decision policy for CW modulation to achieve high throughput performance. In contrast, to address the bandwidth asymmetry challenge in interplanetary links and control the traffic on the reverse channel, at the destination, delayed selective acknowledgement (SACK) CC is employed [5], which maintains a delayed SACK count d_s and sends back one SACK to the source for every d_s data segments received. If there is no segment loss, the destination continues delaying the SACKs with a delayed SACK count d_s. However, it sends a new SACK with an updated block if a segment loss is detected. Subsequently, the source retransmits only the missing data segments.

IV. RL BASED PROBLEM FORMULATION

In this section, we present the formulation of the TCP CC problem as an RL problem in which the objective of the learning agent is to find the optimal decision policy of the CW adjustment based on the challenging and unique interplanetary backhaul link conditions and to maximize the throughput while minimizing the packet loss rate and file transfer time. The RL-based problem formulation involves the concepts of state, action, and reward. From the perspective of the TCP source, the state, action, and reward are defined as follows:

A. State Space

The state of a TCP flow i at an epoch t is defined as follows: $s_t^i = [cw_t^i,\ tp_t^i,\ lr_t^i,\ \Delta rtt_t^i, rtt_t^i,\ \Lambda ack_{t-1,t}^i]$. Here, state parameters cw_t^i, tp_t^i, lr_t^i, $\Delta_{rtt_t^i}$, rtt_t^i, and $\Lambda ack_{t-1,t}^i$ denote the CW, throughput, loss rate, average rate of the change in the RTT, the RTT in epoch t, and the inter-arrival time of the returning ACKs in epochs $t-1$ and t, respectively; cw_t^i denotes the size of the CW for the TCP flow, i, at epoch; and t. tp_t^i calculates the number of data segments transferred successfully from the source to the destination per unit time. The loss rate, lr_t^i, is defined as the ratio of the lost segments to the number of transmitted segments, where the lost segments are calculated by subtracting the number of ACKs received from the total number of transmitted segments. Furthermore, $\Delta_{rtt_t^i}$ denotes the average rate-of-change in the RTT; rtt_t^i corresponds to the interval of the RTT, and $\Lambda ack_{t-1,t}^i$ denotes the inter-arrival time of the returning consecutive ACKs measured at the source at epochs $t-1$ and t. The above-mentioned state parameters are selected to capture the challenging and unique characteristics of interplanetary backhaul links. For instance, the use of state variables rtt_t^i and Δrtt_t^i captures the effects of extremely long and variable propagation delays because measuring the variation in the extremely long RTT accurately may lead to accurate CC behavior. Furthermore, lr_t^i is used to capture high link errors in the interplanetary backhaul links. $\Lambda ack_{t-1,t}^i$ captures frequent link outages and intermittent link connectivity in the connection path between the source and the destination. A small time difference between the returning ACKs implies a high constant rate of the traffic in the connection path. However, the effects of the above-mentioned challenges on the throughput performance can be captured from tp_t^i. Because an end host may simultaneously execute more than one TCP flows, the system state can be defined as $s_t = [s_t^1, \ldots, s_t^i, \ldots, s_t^N]$, where N denotes the total number of TCP flows being executed by an end host.

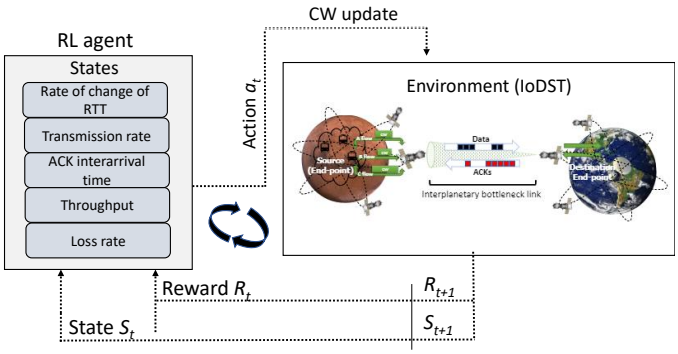


Fig. 3: Schematic of RL for TCPCC in IoDST.

B. Action Space

An action a_t is defined as $a_t = [a_t^1, \ldots, a_t^i, \ldots, a_t^N]$, where a_t^i denotes the action performed on a TCP flow i and N denotes the total number of TCP flows being executed on an end host. The action, a_t^i, specifies the change that needs to be made to the CW of the TCP flow, i, at epoch t. The CW is updated by increasing, decreasing, or retaining its rate of transmission.

C. Reward Function

The reward function yields the network utility of a TCP flow i. Our objective for the reward function is to determine the optimal decision policy for the CW to fully utilize the end-to-end bandwidth without causing network congestion. Thus, we design our reward function based on the bandwidth utilization and the effects of network congestion on the connection path. In this regard, we select the throughput metric to estimate the bandwidth used by a TCP flow i. Moreover, the RTT value and packet loss rate are used to capture the effects of congestion on the connection path. The reward function is defined as follows:

$$R_t = \sum_{i=1}^{N} \log(tp) - \log(lr) - \log(RTT), \tag{1}$$

where tp denotes the throughput of a TCP flow i, lr denotes the number of lost segments, RTT is the interval of the RTT between the source and the destination, and the $\log(\cdot)$ function ensures that each TCP flow shares a reasonable proportion of the network resources. The first term, $\log(tp)$, aims to maximize the throughput while ensuring that the bandwidth of the bottleneck link is reasonably shared among the TCP flows. Because interplanetary backhaul links are characterized by high link errors and long propagation delays, the second term, $\log(lr)$, tends to minimize the packet loss rate and

the third term, $\log(RTT)$, aims to minimize the propagation delays while keeping the RTTs of different flows close to the maximum extent. The reward function incentivizes each TCP flow i to maximize the throughput performance while minimizing both the packet loss rate and RTT.

As shown in Fig. 3, in each decision epoch t, the agent receives some representation of the state of the IoDST environment, based on which it selects an action for the CW modulation of a TCP flow i. Network utility is calculated and used as a reward signal to optimize the decision policy during the next epoch, and the learning agent continues to explore the optimal policy by performing sequential actions (varying the CW) on the feedback received to achieve its desired objective.

V. Deep Reinforcement Learning-Based Congestion Control

In this section, we briefly introduce DRL followed by the OAC-based DRL framework for TCPCC to determine the optimal CW decision policy in the IoDST communication.

A. Deep Reinforcement Learning Approach

In an RL problem, a learning agent interacts with its environment over discrete decision epochs $t_i \in \mathcal{T}$, where $\mathcal{T} = \{t_1, t_2, \ldots, t_n\}$. During each decision epoch t_i, the RL agent receives the information of a state of the environment, $s_{t_i} \in S$, from the state space, $S = \{s_{t_1}, s_{t_2}, \ldots, s_{t_n}\}$, and performs an action $a_{t_i} \in A$, where $A = \{a_{t_1}, a_{t_2}, \ldots, a_{t_n}\}$. Subsequently, the RL agent receives a numerical reward $r_{t_i} \in R$, where $R = \{r_{t_1}, r_{t_2}, \ldots, r_{t_n}\}$, and enters a new state. The RL agent aims to find a policy $\pi(s_t)$ to map each state to a deterministic action, thereby maximizing its discounted cumulative reward

$$R_t = \sum_{t=1}^{T} \gamma^t r(s_t, a_t), \tag{2}$$

where $\gamma \in [0, 1]$ denotes a factor that discounts future rewards [33]. A deep version of RL [34] called DQN uses a deep neural network (DNN) as a function approximator, which accepts the state–action pair, (s_t, a_t), as the input and outputs the corresponding Q-value, $Q(s_t, a_t)$, given by

$$Q(s_t, a_t) = \mathbb{E}[R_t | s_t, a_t], \tag{3}$$

where $\mathbb{E}[\cdot]$ is an expectation, and the action is derived by applying the following commonly used greedy policy:

$$\pi(s_t) = \arg\max_{a_t} Q(s_t, a_t). \tag{4}$$

The target value, y_t, is calculated using the Bellman equation based on the parameters of target network

$$y_t = r(s_t, \mathrm{a}_t) + \gamma max_{\pi(s_{t+1})} Q^{'}(s_{t+1}, \pi(s_{t+1}) | \theta^{Q^{'}}), \tag{5}$$

where $\theta^{Q^{'}}$ is the parameter of the target network used to compute the target. Then, the DQN can be trained by minimizing the loss, $L(\theta^Q)$, where

$$L(\theta^Q) = \mathbb{E}\big[y_t - Q(s_t, \mathrm{a}_t | \theta^Q)\big]^2. \tag{6}$$

Although DQN solves problems with high-dimensional state spaces [34], it is constrained to achieve discrete control on low-dimensional action spaces, and continuous control cannot be trivially extended [35]. A common approach to realize continuous control in RL that can operate over continuous state and action spaces is based on the policy gradient method [35]–[37]. In this context, the authors of [35] introduced the actor–critic method, which leverages both a DNN and the deterministic policy gradient [38] to ensure continuous control in RL. However, the existing actor–critic algorithms for continuous control are limited by their poor sample efficiency. In [21], two phenomena that prevent efficient exploration in actor–critic algorithms were identified: (1) *pessimistic underexploration* and (2) *directional uninformedness*. The former relies on the lower bound approximation of the critic and greedy actor update; thus, informative and useful actions that may improve the estimation of the critic are not explored. In comparison, the latter phenomenon comprises the use of Gaussian policies to sample actions with equal probability in opposite directions from the current mean. However, exploration in both directions is inefficient because portions of the action space corresponding to high densities of the past policies are already been explored. Nevertheless, an efficient exploration requires sampling actions along certain direction.

B. OAC-based DRL for TCP CC

To address the aforementioned problems of the existing actor–critic algorithms, [21] introduced an OAC approach based on the principle of optimism during uncertainty [22]. The OAC approach approximates both the upper and lower confidence bounds on the state–action value function to perform directed exploration using the upper bound while still using the lower bound to avoid effects of overestimation. Hence, we leveraged the OAC algorithm for TCPCC on the interplanetary backhaul links in an IoDST network. The proposed OAC-TCPCC framework on interplanetary backhaul links (OAC-TCPCC) is depicted in Fig. 4. OAC-TCPCC is implemented using an online DRL algorithm that observes the environment state, takes an action, obtains the reward, and updates the model parameters accordingly. The OAC maintains two DNN functions simultaneously: (1) parameterized primary network function and (2) parameterized target network function. The former is used to derive an action corresponding to the current state using the policy, $\pi_\psi(s_t|\theta^\pi)$, and the Q-value, $Q(s_t, a_t|\theta^Q)$. In contrast, the latter function is used to evaluate the Q-value corresponding to the target state–action pair, $Q(s_{t+1}, a_{t+1}|\theta^{Q^{'}})$. Experience replay is used to collect and store the state transition samples into a replay buffer and update the DNN using a mini-batch sampled from the replay buffer, instead of using immediately collected observation sequences. The exploration policy, π_ψ, used by the primary network is different from the target policy, π_ζ, derived by the target network, and is used only to sample the actions from the environment. The exploration policy, π_ψ, maximizes the upper confidence bound of Q^π each time the agent enters a new state, which increases the probability of exploring and executing

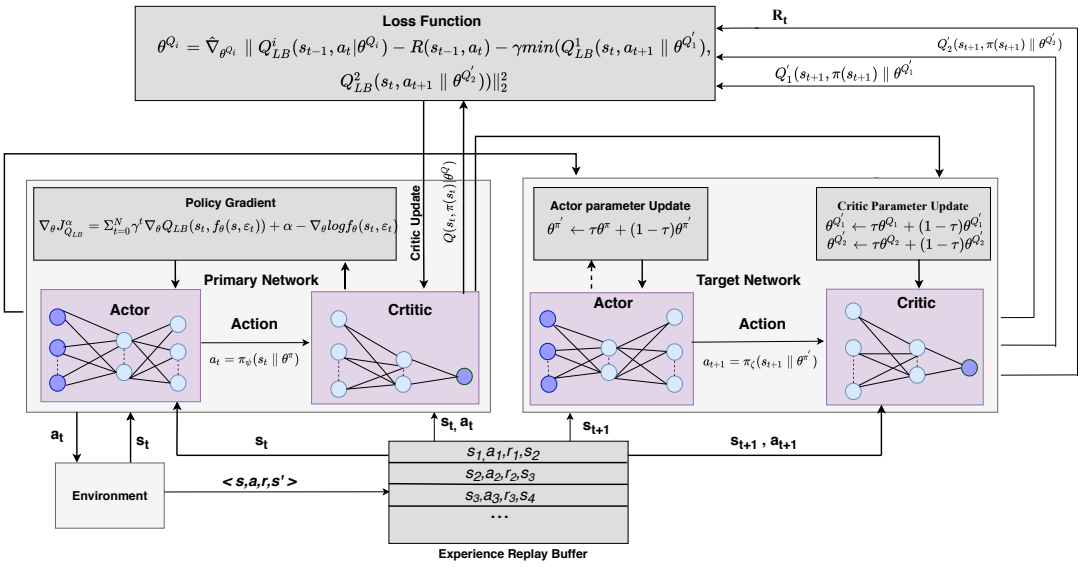


Fig. 4: OAC-TCPCC framework.

informative actions. The exploration policy, $\pi_\psi = \mathcal{N}(\mu_\psi, \chi_\psi)$, is defined as follows [21]:

$$\mu_\psi, \chi_\psi = \underset{\mu,\chi: KL(\mathcal{N}(\mu,\chi), \mathcal{N}(\mu_T,\chi_T)) \leq \delta}{\text{argmax}} \pi_{\psi_{a\sim\mathcal{N}(\mu,\chi)}[\bar{Q}_{UB}(s,a)]}, \tag{7}$$

where μ_ψ is the mean and χ_ψ is the variance of the Gaussian exploration policy. The KL constraint allows to preserve the stability of learning by ensuring that the exploration policy, π_ψ, remains close to the target policy, π_ζ. In addition, $\bar{Q}_{UB}(s,a)$ is the approximate upper bound used by the OAC. The mean, μ_ψ, of the exploration policy, π_ψ, is defined as follows:

$$\mu_\psi = \mu_\zeta + \frac{\sqrt{2\delta}}{\parallel [\nabla_a \hat{Q}_{UB}(s,a)]_{a=\mu_\zeta} \parallel_\chi} \cdot \chi_\zeta [\nabla_a \hat{Q}_{UB}(s,a)]_{a=\mu_\zeta}, \tag{8}$$

where $\chi_\psi = \chi_\zeta$ and μ_ζ denotes the mean of the target policy, π_ζ, and δ denotes the hyperparameter that controls the maximum permissible KL divergence between π_ψ and π_ζ. The term $\sqrt{2\delta}$, affects the average performance of the OAC trained on T decision epochs, and the term $[\nabla_a \hat{Q}_{UB}(s,a)]_{a=\mu_\zeta}$ computes the gradient of the upper bound, $\hat{Q}_{UB}(s,a)$. The approximate upper bound, $\bar{Q}_{UB}(s,a)$, of π_ψ is defined as follows:

$$\bar{Q}_{UB}(s,a) = a^\top \Big[\nabla_a \hat{Q}_{UB}(s,a)\Big]_{a=\mu_T} + \kappa, \tag{9}$$

where $\nabla_a \hat{Q}_{UB}(s,a)$ is the gradient of the upper bound, $\hat{Q}_{UB}(s,a)$. $\hat{Q}_{UB}(s,a) = \mu_Q(s,a) + \beta_{UB}\sigma_Q(s,a)$, where $\mu_Q(s,a)$ and $\beta_{UB}\sigma_Q(s,a)$ are the mean and standard deviation based on the bootstraps of the critic, respectively, $\beta_{UB} \in \mathbb{R}^+$ controls the level of optimism, and κ is a constant.

The exploratory policy, π_ψ, introduced in (7) balances the maximization of the approximate upper bound, $\bar{Q}_{UB}(s,a)$, and constraining of the maximum permissible KL divergence between the exploration policy, π_ψ, and the target policy, π_ζ. This balance preserves the stability of learning and ensures that π_ψ remains within the action range in which the approximate upper bound, $\bar{Q}_{UB}(s,a)$, is accurate. Although the OAC employs an optimistic algorithm, it does not overestimate because the actor and critic updates still follow the lower bound approximation. Thus, the upper bound only influences the critic indirectly by the distribution of the state–action pairs in the experience replay buffer. The actor updates the policy parameter, θ, of π_ψ to maximize the reward, J, by following its gradient [21].

$$\begin{aligned}\nabla_\theta J^\alpha_{Q_{LB}} =& \Sigma^N_{t=0}\gamma^t \nabla_\theta Q_{LB}(s_t, f_\theta(s,\varepsilon_t)) \\ &+ \alpha - \nabla_\theta \log f_\theta(s_t, \varepsilon_t).\end{aligned} \tag{10}$$

The OAC uses a reparameterization policy gradient in which the random variable, ε, is sampled from a standard multivariate Gaussian distribution $\varepsilon \sim \mathcal{N}(0, I)$. The reparameterization function, f, is defined to ensure that the probability density of the random variable, $f(s,\varepsilon)$, is identical to the density of $\pi_\zeta(a \mid s)$. The critic network is updated with two bootstraps of the critic using the lower bound [21].

$$\begin{aligned}\theta^{Q_i} =& \hat{\nabla}_{\theta^{Q_i}} \parallel Q^i_{LB}(s_{t-1}, a_t|\theta^{Q_i}) - r(s_{t-1}, a_t) \\ &- \gamma \min(Q^1_{LB}(s_t, a_{t+1}|\theta^{Q'_1}), Q^2_{LB}(s_t, a_{t+1}|\theta^{Q'_2})) \parallel^2_2 .\end{aligned} \tag{11}$$

The target network parameters for the actor network, $\theta^{Q'_1}$, and the critic network, $\theta^{Q'_2}$, are updated after each interval, τ, using the following equations:

$$\theta^{Q'_1} \leftarrow \tau\theta^{Q_1} + (1-\tau)\theta^{Q'_1}, \tag{12}$$

$$\theta^{Q'_2} \leftarrow \tau\theta^{Q_2} + (1-\tau)\theta^{Q'_2}. \tag{13}$$

We formally present the proposed OAC-TCPCC for interplanetary backhaul links in IoDST in Algorithm 1. First, the algorithm randomly initializes the primary actor network, $\pi(\cdot)$, and the primary critic network, $Q(\cdot)$, with parameters, θ^π and

Algorithm 1 OAC-TCPCC algorithm for interplanetary backhaul network links in IoDST communication

1: Parameters: θ^{π}, θ^{Q_1}, θ^{Q_2}
2: Randomly initialize primary actor $\pi(\cdot)$ and critic network $Q(\cdot)$ with parameters θ^{π}, θ^{Q_1}, θ^{Q_2}
3: Initialize target actor network $\pi^{'}(\cdot)$ and critic network $Q^{'}(\cdot)$ with parameters $\theta^{\pi^{'}} \leftarrow \theta^{\pi}$, $\theta^{Q_1^{'}} \leftarrow \theta^{Q_1}$, $\theta^{Q_2^{'}} \leftarrow \theta^{Q_2}$ and replay buffer $\mathcal{B} \leftarrow \phi$
4: Initialize replay buffer $\mathcal{B}$ with finite size of B
5: **for** episode n = 1, . . ., N **do**
6: Receive initial state s_t
7: Size of CW
8: Number of data segments transferred successfully divided by time
9: Number of lost segments
10: Average rate-of-change in RTT
11: Inter-arrival time of returning ACKs
12: RTT for time t
13: **for** t = 1,2, . . ., T **do**
14: Derive action $a_t \leftarrow \pi_E(a_t|s_t)$ from (10)
15: Execute action a_t
16: Observe reward r_t and next state s_{t+1}
17: Store transition experience $\mathcal{B} \leftarrow \mathcal{B}\ (s_t, a_t, r_t, s_{t+1})$
18: Sample random mini-batch of $\mathcal{B}'$ transitions
19: **for** each training step **do**
20: **for** $i \in$ {1,2} **do**
21: update θ^{Q_i} with $\hat{\nabla}_{\theta^{Q_i}} \|Q^i_{LB}(s_t, a_t|\theta^{Q_i}) - r\ (s_t, a_t) - \gamma\ \min(Q^1_{LB}\ (s_{t+1}, a_{t+1}|\theta^{Q_1^{'}}), Q^2_{LB}(s_{t+1}, a_{t+1}|\theta^{Q_2^{'}}))\|^2_2$
22: **end for**
23: update θ^{π} with $\nabla_{\theta} J^{\alpha}_{Q_{LB}}$
24: $\theta^{Q_1^{'}} \leftarrow \tau\theta^{Q_1} + (1-\tau)\theta^{Q_1^{'}}$;
25: $\theta^{Q_2^{'}} \leftarrow \tau\theta^{Q_2} + (1-\tau)\theta^{Q_2^{'}}$;
26: **end for**
27: **end for**
28: **end for**
29: **end**

θ^{Q_1}, θ^{Q_2}, respectively. The target actor network, $\pi^{'}(\cdot)$, and the target critic network, $Q^{'}(\cdot)$, share the same network structure as the primary actor and critic networks and are initialized with parameters, $\theta^{\pi^{'}}$ and $\theta^{Q_1^{'}}$, $\theta^{Q_2^{'}}$, respectively. The target networks are used to improve the learning stability and are gradually updated using the control parameter, τ. Because the parameters of the actor and critic networks are initialized randomly, our algorithm does not depend on the decision policy derived by the actor network and performs sufficient exploration using random transition samples to obtain the necessary knowledge of positive and negative experiences to finally learn the optimal policy. The OAC-TCPCC agent receives the initial state of the network consisting of the size of CW, number of data segments transferred successfully divided by time, number of lost segments, average rate-of-change in RTT, inter-arrival time of returning ACKs, and the RTT for time t (lines 6-11), and the action for CC, e.g., the change of CW of a TCP flow i, is derived from the actor network using the exploration policy, $\pi_E(a_t|s_t)$ (line 14). The exploration policy, $\pi_E(a_t|s_t)$ (line 14) is an optimistic estimate for continuous control and is computed each time the agent enters a new state. Since the change of CW has a significant impact on throughput, a reward is received (line 16) that aims to maximize the throughput while ensuring that the bandwidth of the bottleneck link is reasonably shared among the TCP flows. Because interplanetary backhaul links are characterized by high link errors and long propagation delays, our reward function tends to minimize the packet loss rate and the propagation delays while keeping the RTTs of different flows close to the maximum extent. The experience replay buffer stores the transition samples (line 17) and subsequently randomly samples them into a mini-batch of $\mathcal{B}'$ samples to train the actor and critic networks. Finally, the primary critic network is updated with two bootstraps of the critic using the lower bound (line 21). The primary actor network is also updated using a lower bound from the chain rule defined in (10) (line 23), followed by the soft updates of the target actor and critic networks (lines 24 and 25).

TABLE I: Simulation Parameters

Parameters	Value
RTT	8.5 min–40 mins [5]
Bandwidth	1 Mbps [6]
Learning rate of actor, LRA	$3e^{-3}$
Learning rate of critic, LRC	$3e^{-3}$
Reward discount factor, γ	0.95
Replay buffer size, B	10^6
Mini-batch size, $\mathcal{B}'$	512
Number of hidden layers (both actor and critic networks)	2
Number of neurons per hidden layer (both actor and critic networks)	256
Activation function for hidden layer	ReLU
Activation function for output layer	Identity
Number of training episodes, N	2x10^4
Number of time steps in each episode, T	5x10^3
Optimizer	Adam Optimizer
Optimism level, β_{UB}	4.46 [21]
Delayed SACK factor, d_s	5 [5]
Segment loss probability, p	10^{-5}–10^{-1} [6]
Target transmission rate	100 KB [5]
Data segment size	1 KB
ACK size	40 B
Sink buffer size	100 segments
Size of data file	1 MB–200 MB [5]

VI. Numerical Simulations

To evaluate the performance of the proposed OAC-TCPCC algorithm, we conducted extensive computer simulation experiments. First, the simulation setup is described. Subsequently, convergence performance of the proposed algorithm is discussed. Following that, the evaluation of the throughput performance as a function of the RTT and the file size for different values of segment loss probability is presented. Finally, the proposed OAC-TCPCC algorithm is compared with several existing CC algorithms in terms of the throughput, file transfer time, and fairness.

A. Simulation Setup

We built an IoDST network model consisting of a planetary gateway spacecraft orbiting around an outer space planet,

e.g., Mars, (considered as a source), and a planetary gateway spacecraft orbiting around the Earth planet (considered as a destination), respectively, as shown in Fig. 2. We developed the proposed OAC-TCPCC algorithm using a more practical and realistic channel model, which is also reflected in national aeronautics and space administration (NASA) space communications [39]–[41] and has been implemented in network simulator (NS-3) [42] and PyTorch with Python 3.8 on Ubuntu 20.04 LTS. The simulations were implemented on a PC powered by Intel(R) Core(TM) i7-10700F CPU with frequency 2.9 GHz and RAM 32 GB. The graphics card was NVIDIA GeForce RTX2070 SUPER with an 8-GB memory. Prior to these simulation experiments, we constructed fully connected DNNs for the parameterized primary network and the parameterized target network with two hidden layers for each network, and the number of neurons in the corresponding layers were set to [256, 256] and [256, 256], respectively. We used the Identity function as the activation function for the output layer of the actors and the ReLU function as the activation function of all the hidden layers. We used the experience reply memory and the reply buffer size and batch size is set to 10^6 and 512, respectively. The maximum number of training episodes and the time steps per episode are set to $2\text{x}10^4$ and $5\text{x}10^3$. The learning rates for primary actor network (LRA) and critic network (LRC) are set to (LRA, LRC) = $(3e-3,\ 3e-3)$, $(3e-4,\ 3e-3)$, and $(3e-4,\ 3e-4)$. The reward discount factor, γ, is set to γ = 0.95, γ = 0.85, and γ = 0.5. The OAC-TCPCC simulation experiments were conducted with respect to the transmission of files of varying sizes between 1 MB and 200 MB on very short (8.5-min), long (20-min), and very long (40-min) RTT links. The source and destination are connected through 1 Mbps link, which is assumed to have a packet loss probability, p, of 10^{-5}–10^{-1}. The target transmission rate of the OAC-TCPCC source is assumed to be 100 KB for data segments of size 1 KB. The delayed SACK factor is set to be 5 packets for ACK packets of size 40 KB. The simulation parameters are based on the studies in [5], [6] [21]. The key parameters of the simulations are summarized in Table I.

Our proposed OAC-TCPCC algorithm is compared with six baseline CC schemes, detailed as follows:

- *TP-Planet:* A transport protocol for data traffic in interplanetary Internet that is based on a rate-based AIMD CC to help avoid throughput degradation [5].
- *TCP Peach:* A CC scheme that is based on the use of dummy segments for improving the goodput performance in satellite networks [24].
- *FAST TCP:* A delay-based approach that addresses the CC at large window sizes for high-speed long-latency networks [23].
- *TCP CUBIC:* A CC scheme that modifies the linear CW growth function of existing TCP protocols to be a cubic function to improve the scalability of TCP over fast and long distance networks [26].
- *TCP BBR:* It periodically estimates the available bandwidth and minimal RTT and determines the CW to approach Kleinrocks's optimal operating point [25], [30].

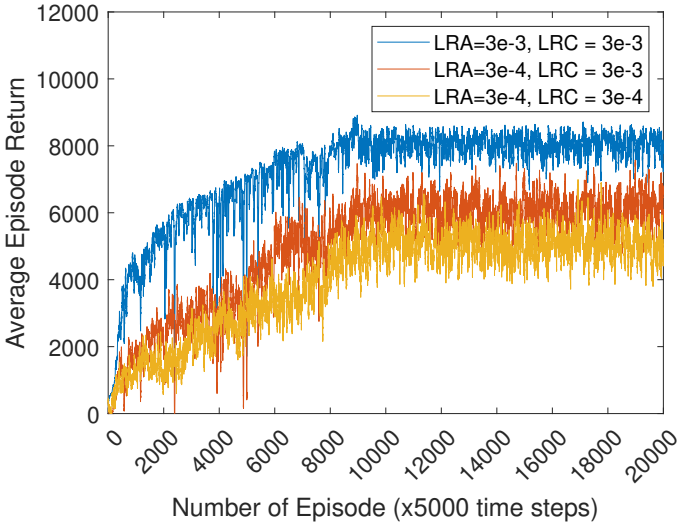


Fig. 5: Convergence performance of proposed algorithm with different learning rates. (LRA: learning rate of actor; LRC: learning rate of critic)

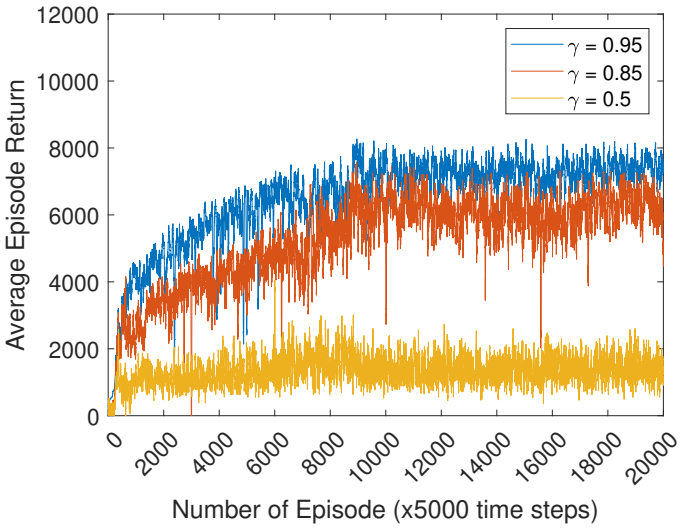
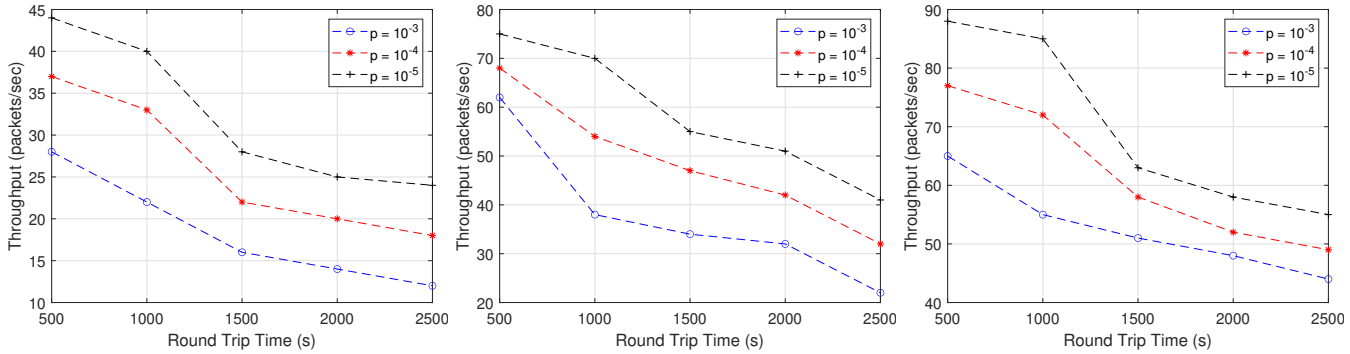


Fig. 6: Convergence performance of proposed algorithm with different reward discount factor rates.

- *DRL-CC:* A DRL-based CC in multi-path TCP (MPTCP) which aims to maximize overall utility (such as goodput performance) [17].

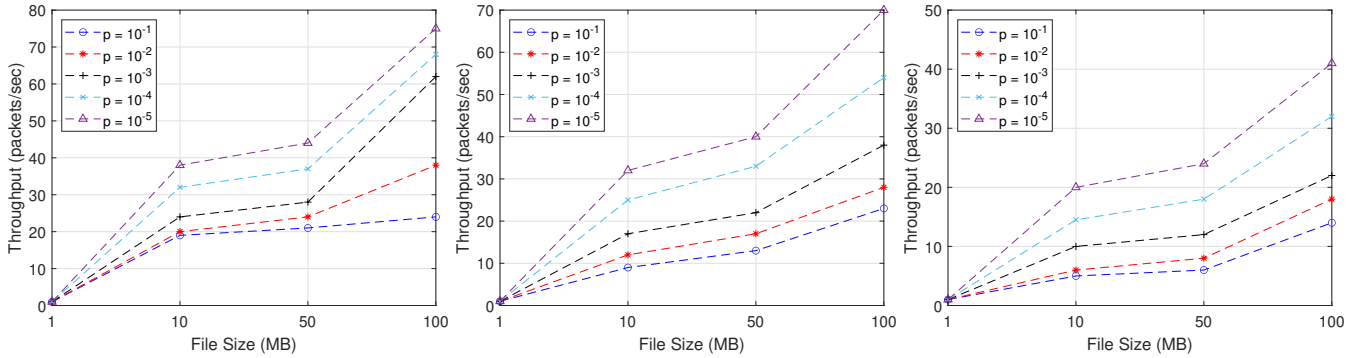
B. Convergence Performance

Fig. 5 shows the convergence performance of our proposed OAC-TCPCC algorithm with various learning rates (LRs) of the primary actor and critic networks. The LR of the actor (LRA) and LR of the critic (LRC) are used by the Adam optimizer for updating weights θ^{π} and θ^{Q_1}, θ^{Q_2} of the primary actor and critic networks, respectively. The number of training episodes and training steps in each episode are N = 20000 and T = 5000, respectively. Based on Fig. 5, the average episode return (total reward) increases as the number of training episodes increases until it becomes relatively stable



(a) 50-MB file size (b) 100-MB file size (c) 200-MB file size

Fig. 7: Throughput performance for different RTTs with segment loss probabilities p = 10^{-5}, 10^{-4}, and 10^{-3} and file sizes = 50 MB, 100 MB, and 200 MB.



(a) 8.5-min RTT (b) 20-min RTT (c) 40-min RTT

Fig. 8: Throughput performance for different file sizes with segment loss probabilities p = 10^{-5}, 10^{-4}, 10^{-3}, 10^{-2}, and 10^{-1} and RTTs = 8.5 min, 20 min, and 40 min.

and converges within a specific range. Moreover, the algorithm converges at approximately 10000 episodes for the different learning rates. This is because the OAC-TCPCC algorithm conducts optimistic exploration by applying the principle of optimism during uncertainty, which increases the probability of executing more informative actions. The average episode return achieved with (LRA, LRC) = $(3e-3,\ 3e-3)$ is significantly higher than those with (LRA, LRC) = $(3e-4,\ 3e-3)$ and (LRA, LRC) = $(3e-4,\ 3e-4)$. Therefore, we select the training result with (LRA, LRC) = $(3e-3,\ 3e-3)$ for evaluating the remaining performance comparison of the proposed OAC-TCPCC algorithm.

Fig. 6 shows the impact of the reward discount factor, γ, on the convergence performance of the proposed algorithm for various values of the discount factor: γ = 0.95, γ = 0.85, and γ = 0.5. A small value of the discount factor indicates that the agent learns from the immediate and near future rewards. In contrast, a large value of the discount factor indicates that the agent focuses more on the long-term reward. As shown in Fig. 6, the convergence performance of the proposed algorithm improves, and the highest average episode return is achieved with reward discount factor value γ = 0.95 because the learning agent considers more the reward the next state will obtain. Therefore, we select γ = 0.95 for evaluating the remaining performance comparisons of the proposed algorithm.

C. Performance Evaluation

We save weights θ^{π}, and θ^{Q_1}, θ^{Q_2} of the primary actor and critic networks that provide the highest average return for determining the optimal CW policy for TCPCC in IoDST environments. To study the throughput performance of the proposed OAC-TCPCC algorithm on the interplanetary backhaul links, we conducted several simulation experiments by varying the packet loss probability and file sizes. Fig. 7 shows the average throughput performance of the proposed OAC-TCPCC algorithm with varying RTTs and segment loss probabilities ranging from 10^{-5} to 10^{-3} for the transmission of files of various sizes. Fig. 7(a) shows that the throughput performance is high when the RTT is short and the segment loss probability is low, and it begins to decrease as the propagation delay and segment loss probability increase. However, increase in the file size improves the throughput performance, as shown in Figs. 7(b) and 7(c). Based on Fig. 7(c), the throughput performance increases up to 87.5 packets/s and approaches the optimal transmission rate for the short RTT and a segment loss probability of 10^{-5} during the transmission of a 200-MB file size. This is because the OAC-TCPCC algorithm detects the changes in the IoDST network conditions during

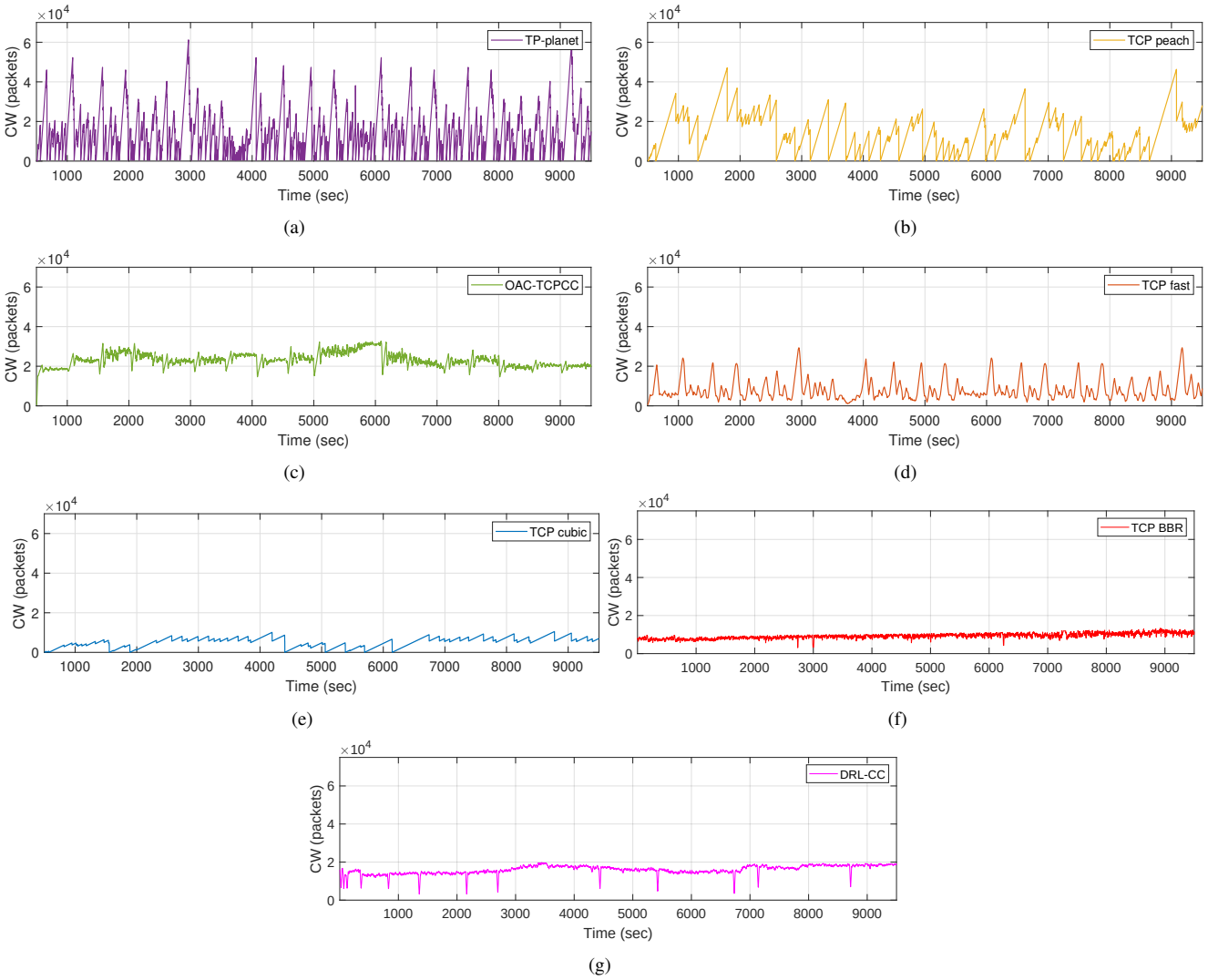


Fig. 9: Evolution of CW versus time with packet loss probability, $p = 10^{-3}$ and RTT = 8.5 min for OAC-TCPCC algorithm and existing CC algorithms.

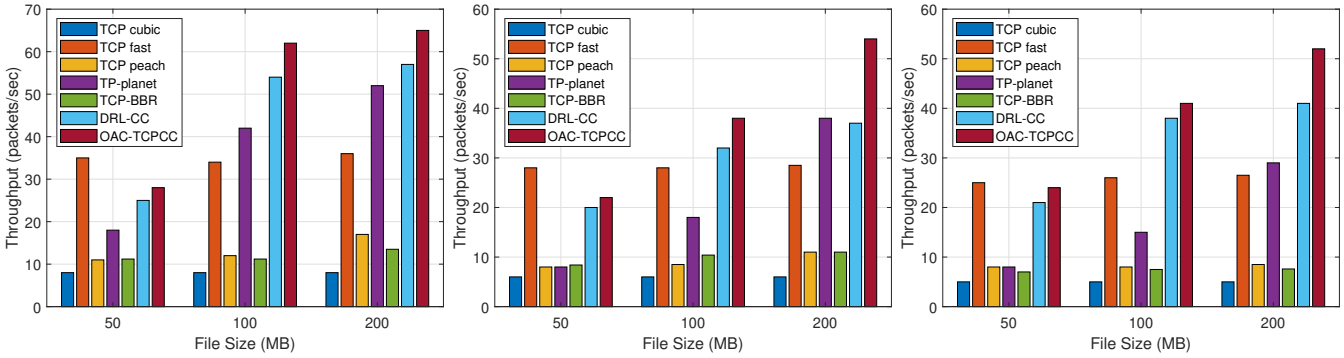
the lifetimes of the flows and learns the appropriate CW policy by maximizing the expected cumulative discounted reward.

Fig. 8 shows the average throughput performance of the proposed algorithm with varying file sizes and segment loss probabilities ranging from 10^{-5} to 10^{-1} for different RTTs. From Fig. 8(a), short RTT links between pairs of IoDST sources and destinations correspond to high throughput performances for the transmission of files of various sizes. However, as the file size increases, the throughput performance improves because of the improved spectral efficiency. As illustrated in Figs. 8(b) and 8(c), increase in the RTT degrades the throughput performance for various segment loss probabilities and file sizes. This decrease in the performance is primarily caused by the adverse effects of high link errors on the links with long propagation delays, as the TCP source retransmits the lost segments on high error links.

D. Performance Comparison

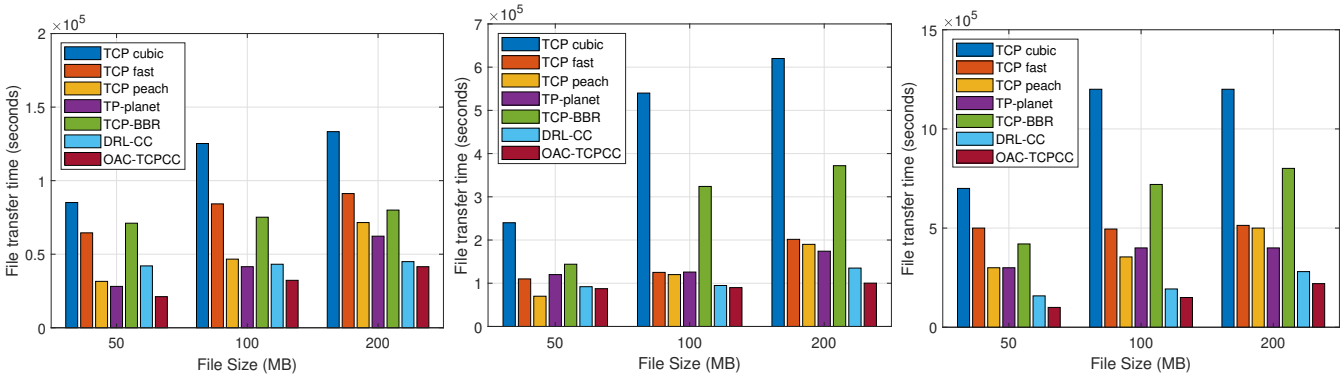
We conducted simulation experiments considering various simulation parameters and evaluated the performance of OAC-TCPCC, as compared with other baseline schemes. Fig. 9 shows the comparison of change of CW size as a function of time. Fig. 9(a) shows that TP-Planet increases the CW rapidly by half the rate of our proposed OAC-TCPCC algorithm, depicted in Fig. 9(c). However, TP-Planet cannot stably control its CW rate. Our proposed OAC-TCPCC algorithm achieves better CW stability than TP-Planet, as shown in Fig. 9(c). Although, DRL-CC achieves CW stability compared with other baselines. It also shows lower CW update performance compared to OAC-TCPCC, as shown in Fig. 9(g). TCP-Peach also captures the link resources rapidly. However, it is quite inefficient in controlling its CW rate during the connection. In contrast, FAST TCP, TCP BBR, and TCP CUBIC show very poor performance on CW update in interplanetary backhaul networks owing to their rule-based CC mechanism.

Fig. 10 illustrates throughput performance comparison with



(a) 8.5-min RTT (b) 20-min RTT (c) 40-min RTT

Fig. 10: Throughput performance comparison for different file sizes with segment loss probability $p = 10^{-3}$ and RTTs = 8.5 min, 20 min, and 40 min.



(a) 8.5-min RTT (b) 20-min RTT (c) 40-min RTT

Fig. 11: File transfer time comparison for different file sizes with segment loss probability $p = 10^{-3}$ and RTTs = 8.5 min, 20 min, and 40 min.

respect to different file sizes and a segment loss probability of 10^{-3} on different RTT links. For this evaluation, we varied file sizes from 50 MB to 200 MB. As is apparent from Fig. 10, TCP CUBIC shows very poor performance on the interplanetary backhaul links. It achieves a throughput of only 8.5 packets/s even on short RTT links, as shown in Figs. 10(a). The performance degradation increases with increasing file sizes and RTT. This is because the interplanetary backhaul links are characterized by high link error rates, and TCP CUBIC increases CW using the concave profile of a cubic function after a window reduction following a loss event. TCP BBR also performs very poorly on interplanetary backhaul links. It achieves a throughput of only 11 packets/s even for a file size of 50 MB on short RTT links, as shown in Fig. 10(a). As RTT increases, TCP BBR results in throughput degradation as shown in Figs. 10(b) & 10(c). TCP Peach achieves a throughput of 17 packets/s for a file size of 200 MB even on short RTT links. However, as the RTT increases, TCP Peach results in performance degradation, as shown in Figs. 10(b) and 10(c). Although, TP-Planet improves throughput for the transmission of large file size on short RTT links, Figs. 10(b) and 10(c) present that the throughput performance decreases with increasing RTT. However, this performance degradation is not severe compared to that with TCP Peach, TCP BBR, and TCP CUBIC. FAST TCP achieves higher throughput of about 25% for small file size, e.g., 50. However, it increases file transfer time by up to 75% under the same conditions (e.g., 50 MB file size and 8.5 min RTT), as shown in Fig. 11(a). In addition, FAST TCP decreases throughput performance with increasing file sizes and long RTT links, as shown in Figs. 10(b) & 10(c). This is because FAST TCP decreases CW gradually and then rapidly as the delay increases. However, the proposed OAC-TCPCC learns intelligent CC to maximize throughput while minimizing file transfer time. It achieves a considerably good throughput performance for both long and short TCP flows on short, long, and very long RTT links. We also calculate average reward for proposed OAC-TCPCC and baseline CC approaches for different link RTTs and file sizes with segment loss probability of 10^{-3}. The corresponding results are shown in Table II. It can be observed from Table II that the average reward of OAC-TCPCC outperforms other baselines. In other words, OAC-TCPCC is more robust to varying file sizes and link RTTs compared with baselines, improving the throughput while reducing file transfer time.

We also compare the performance with a DRL-based CC scheme called DRL-CC. As shown in Fig. 10, DRL-CC improves throughput performance compared to traditional rule-

TABLE II: Average Reward of OAC-TCPCC and Baseline CC Schemes for Different Values of RTT and File Size

RTT	File Size	TCP CUBIC	FAST TCP	TCP Peach	TP-Planet	TCP BBR	DRL-CC	OAC-TCPCC (Best)
8.5 min	50 MB	3.154	10.5448	5.6642	8.0548	5.1054	9.2882	11.8254
	100 MB	2.11584	10.8662	6.72	9.1885	5.1034	12.448	15.85
	200 MB	2.0154	11.224	7.1411	9.4856	5.1062	12.6652	15.9422
20 min	50 MB	2.79384	9.88105	4.0235	4.6325	4.9822	8.49982	10.5432
	100 MB	2.79054	9.998	4.1055	7.51465	4.984	9.4356	13.446
	200 MB	2.79992	10.31	4.1307	10.54	4.9861	9.9205	14.8341
40 min	50 MB	2.5411	8.4358	3.9842	4.5882	3.9254	8.3054	10.4826
	100 MB	2.50554	8.4934	3.983	5.69982	3.919	9.2367	12.6481
	200 MB	2.48	8.5105	3.984	8.6154	3.904	9.8443	14.5614

based approaches. This is attributed to the fact that DRL-CC characterizes network features in dynamic environments. However, the proposed OAC-TCPCC algorithm outperforms existing CC algorithms by achieving highest throughput performance of 66 packets/s, 54 packets/s, and 52 packets/s for a file size of 200 MB on short, long, and very long RTT links, as shown in Fig. 10(a)–10(c), respectively. In Fig. 10(c), for the transmission of 200 MB file, OAC-TCPCC enhances throughput performance by 766%, 643%, 477%, 92%, 79%, and 27% compared to TCP CUBIC, TCP BBR, TCP Peach, FAST TCP, TP-Planet, and DRL-CC, respectively. This is attributed to the employment of an intelligent CC employing an OAC-based DRL algorithm that achieves sample efficiency in challenging IoDST environments where obtaining data samples is critical.

Fig. 11 presents file transfer time of the proposed OAC-TCPCC, DRL-CC, TP-Planet, TCP Peach, FAST TCP, TCP BBR and TCP CUBIC on varying RTT links with respect to different file sizes and a segment loss probability of 10^{-3}. As shown in Fig. 11(a), the transfer of a file of size 200 MB takes approximately 666 min on short RTT link (e.g., 8.5 min) using the proposed OAC-TCPCC algorithm. However, it takes approximately 1,483 min, 1,383 min, and 1,333 min using TCP CUBIC, FAST TCP, and TCP BBR respectively. Although, TCP Peach, TP-Planet, and DRL-CC decrease file transfer delay, the reductions are only 47%, 32%, and 11%, respectively. Fig. 11(a) shows that OAC-TCPCC algorithm outperforms by reducing the file transfer time by up to 55%, 51%, 50%, 47%, 32%, and 11% compared to TCP CUBIC, FAST TCP, TCP BBR, TCP Peach, TP-Planet, and DRL-CC, respectively. Figs. 11(b) and 11(c) depict file transfer time comparison with respect to varying file sizes corresponding to propagation delays of 20 min and 40 min, respectively. An increase in the propagation delay increases file transfer delay. In addition, the delay increases with the increase in file size. As the file size is increased to 200 MB for transmission on the link of 40 min long RTT, the proposed OAC-TCPCC scheme realizes a delay reduction by up to approximately 79%, 58%, 50%, 48%, 37%, and 16% compared to TCP CUBIC, TCP BBR, FAST TCP, TCP Peach, TP-Planet, and DRL-CC respectively. We also compared the fairness between different TCP flows for the proposed OAC-TCPCC algorithm and the existing CC algorithms. Fig. 12 shows the Jain fairness index as a function of the number of TCP flows when the bottleneck capacity is 1 Mbps. Our proposed OAC-TCPCC algorithm keeps a high fairness rate for bandwidth allocations to multiple TCP flows. This is because we designed our reward function to achieve a good trade-off between fair bandwidth allocations and throughput.

VII. Conclusion and Future Work

In an IoDST system, large volumes of real-time space-related information and control data are generated, processed, and exchanged by various spacecrafts over the interplanetary backhaul network. This paper presented the design and evaluation of an intelligent CC employing an OAC-DRL framework called OAC-TCPCC for the IoDST networks. Our proposed OAC-TCPCC algorithm addresses the issue of the low throughput performance of the existing TCP schemes in interplanetary backhaul links owing to their inefficient window-based CC mechanisms. OAC-TCPCC adjusts the CW intelligently to address the highly variable network conditions in IoDST environments, thereby mitigating the degradation in the throughput performance. Simulation results demonstrate that the proposed scheme outperforms existing inefficient window-based CC schemes in terms of the throughput performance as well as file transfer time and fairness improvement.

Despite the promising experimental results show that OAC-TCPCC is capable of learning intelligent CC policies and improves TCP throughput performance and reduces delay than the state-of-the-art TCP CC approaches, we identified challenges of the OAC-TCPCC to be addressed in the future research. For instance, OAC-TCPCC consists of several tunable hyperparameters. Searching for the optimized hyperparameter is a challenging task that requires future efforts to optimize learning models with fewer hyperparameters [43]. Moreover, DL frameworks such as meta-learning (ML) can be applied to transfer knowledge among learning models and train the model to deal with unseen scenarios [44], [45]. Although OAC-TCPCC framework could also be adapted to applications with similar challenges as IoDST networks (e.g., disruptive-tolerant networks (DTNs), we will investigate the performance of our proposed OAC-TCPCC approach against DTNs in our future work. Furthermore, we will evaluate the performance of

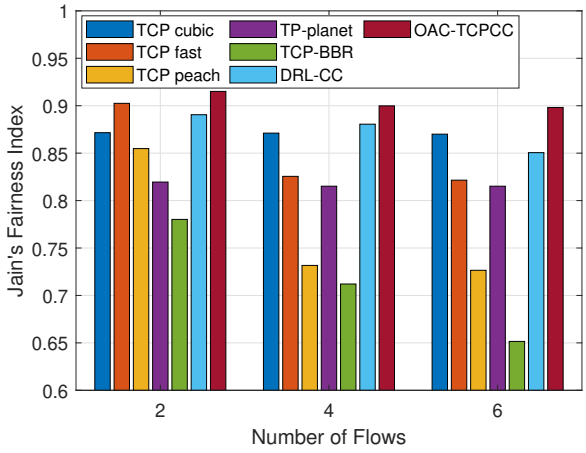


Fig. 12: Comparison of fairness index versus number of TCP flows for OAC-TCPCC algorithm and existing CC algorithms.

the proposed OAC-TCPCC by performing emulated tests using network emulator that would be capable of testing network applications in deep space communications [46], [47].

REFERENCES

[1] I. F. Akyildiz, Ö. B. Akan, C. Chen, J. Fang, and W. Su, "Interplanetary Internet: state-of-the-art and research challenges," *Computer Networks*, vol. 43, no. 2, pp. 75–112, 2003.

[2] L. Xing, "Reliability in Internet of things: Current status and future perspectives," *IEEE Internet of Things Journal*, vol. 7, no. 8, pp. 6704–6721, 2020.

[3] N.-N. Dao, D.-N. Vu, A. Masood, W. Na, and S. Cho, "Reliable broadcasting for safety services in dense infrastructureless peer-aware communications," *Reliability Engineering & System Safety*, vol. 193, p. 106655, 2020.

[4] K. Bhasin and J. L. Hayden, "Space Internet architectures and technologies for NASA enterprises," *International Journal of Satellite Communications*, vol. 20, no. 5, pp. 311–332, 2002.

[5] O. B. Akan, J. Fang, and I. F. Akyildiz, "TP-Planet: A reliable transport protocol for interplanetary Internet," *IEEE Journal on Selected Areas in Communications*, vol. 22, no. 2, pp. 348–361, 2004.

[6] O. B. Akan, H. Fang, and I. F. Akyildiz, "Performance of TCP protocols in deep space communication networks," *IEEE Communications Letters*, vol. 6, no. 11, pp. 478–480, 2002.

[7] L. Grieco and S. Mascolo, "A congestion control algorithm for the planetary Internet," *IFAC Proceedings Volumes*, vol. 38, no. 1, pp. 119–124, 2005.

[8] L. A. Grieco and S. Mascolo, "A congestion control algorithm for the deep space Internet," *Space Communications*, vol. 20, no. 3-4, pp. 155–160, 2006.

[9] I. Psaras, G. Papastergiou, V. Tsaoussidis, and N. Peccia, "DS-TP: Deep-space transport protocol," in *2008 IEEE Aerospace Conference*. IEEE, 2008, pp. 1–13.

[10] G. Papastergiou, I. Psaras, and V. Tsaoussidis, "Deep-space transport protocol: a novel transport scheme for space DTNs," *Computer Communications*, vol. 32, no. 16, pp. 1757–1767, 2009.

[11] X. Wei, J. Zhao, L. Zhou, and Y. Qian, "Broad reinforcement learning for supporting fast autonomous IoT," *IEEE Internet of Things Journal*, vol. 7, no. 8, pp. 7010–7020, 2020.

[12] K. Winstein and H. Balakrishnan, "TCP ex machina: Computer-generated congestion control," *ACM SIGCOMM Computer Communication Review*, vol. 43, no. 4, pp. 123–134, 2013.

[13] W. Li, F. Zhou, K. R. Chowdhury, and W. M. Meleis, "QTCP: Adaptive congestion control with reinforcement learning," *IEEE Transactions on Network Science and Engineering*, 2018.

[14] K. Xiao, S. Mao, and J. K. Tugnait, "TCP-Drinc: Smart congestion control based on deep reinforcement learning," *IEEE Access*, vol. 7, pp. 11 892–11 904, 2019.

[15] L. Cui, Z. Yuan, Z. Ming, and S. Yang, "Improving the congestion control performance for mobile networks in high-speed railway via deep reinforcement learning," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 6, pp. 5864–5875, 2020.

[16] J. Xu, B. Ai, L. Wu, and L. Chen, "Handover-aware cross-layer aided tcp with deep reinforcement learning for high-speed railway networks," *IEEE Networking Letters*, vol. 3, no. 1, pp. 31–35, 2020.

[17] Z. Xu, J. Tang, C. Yin, Y. Wang, and G. Xue, "Experience-driven congestion control: When multi-path TCP meets deep reinforcement learning," *IEEE Journal on Selected Areas in Communications*, vol. 37, no. 6, pp. 1325–1336, 2019.

[18] M. Chen, R. Li, J. Crowcroft, J. Wu, Z. Zhao, and H. Zhang, "Ran information-assisted tcp congestion control using deep reinforcement learning with reward redistribution," *IEEE Transactions on Communications*, 2021.

[19] W. Li, S. Gao, X. Li, Y. Xu, and S. Lu, "Tcp-neuroc: Neural adaptive tcp congestion control with online changepoint detection," *IEEE Journal on Selected Areas in Communications*, vol. 39, no. 8, pp. 2461–2475, 2021.

[20] S. Abbasloo, C.-Y. Yen, and H. J. Chao, "Wanna make your tcp scheme great for cellular networks? let machines do it for you!" *IEEE Journal on Selected Areas in Communications*, vol. 39, no. 1, pp. 265–279, 2020.

[21] K. Ciosek, Q. Vuong, R. Loftin, and K. Hofmann, "Better exploration with optimistic actor critic," in *Proceedings of Advances in Neural Information Processing Systems*, Vancouver, Canada, 8–12 Dec 2019, pp. 1787–1798.

[22] B. D. Ziebart, "Modeling purposeful adaptive behavior with the principle of maximum causal entropy," 2010.

[23] D. X. Wei, C. Jin, S. H. Low, and S. Hegde, "FAST TCP: Motivation, architecture, algorithms, performance," *IEEE/ACM transactions on Networking*, vol. 14, no. 6, pp. 1246–1259, 2006.

[24] I. F. Akyildiz, G. Morabito, and S. Palazzo, "TCP-Peach: a new congestion control scheme for satellite IP networks," *IEEE/ACM Transactions on Networking*, vol. 9, no. 3, pp. 307–321, 2001.

[25] N. Cardwell, Y. Cheng, C. S. Gunn, S. H. Yeganeh, and V. Jacobson, "Bbr: congestion-based congestion control," *Communications of the ACM*, vol. 60, no. 2, pp. 58–66, 2017.

[26] S. Ha, I. Rhee, and L. Xu, "CUBIC: a new TCP-friendly high-speed TCP variant," *ACM SIGOPS operating systems review*, vol. 42, no. 5, pp. 64–74, 2008.

[27] L. A. Grieco and S. Mascolo, "Performance evaluation and comparison of Westwood+, New Reno, and Vegas TCP congestion control," *ACM SIGCOMM Computer Communication Review*, vol. 34, no. 2, pp. 25–38, 2004.

[28] H. Balakrishnan, V. N. Padmanabhan, S. Seshan, and R. H. Katz, "A comparison of mechanisms for improving tcp performance over wireless links," *IEEE/ACM Transactions on Networking*, vol. 5, no. 6, pp. 756–769, 1997.

[29] R. Wang, T. Taleb, A. Jamalipour, and B. Sun, "Protocols for reliable data transport in space Internet," *IEEE Communications Surveys & Tutorials*, vol. 11, no. 2, pp. 21–32, 2009.

[30] L. Kleinrock, "Internet congestion control using the power metric: Keep the pipe just full, but no fuller," *Ad hoc networks*, vol. 80, pp. 142–157, 2018.

[31] K. Scott and S. Burleigh, "Bundle protocol specification," 2007.

[32] M. Ramadas, S. Burleigh, S. Farrell *et al.*, "Licklider transmission protocol-specification," *IETF request for comments RFC*, vol. 5326, 2008.

[33] R. S. Sutton and A. G. Barto, "Reinforcement learning: An introduction," 2011.

[34] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski *et al.*, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[35] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning," *arXiv preprint arXiv:1509.02971*, 2015.

[36] R. S. Sutton, D. A. McAllester, S. P. Singh, and Y. Mansour, "Policy gradient methods for reinforcement learning with function approximation," in *Proceedings of Advances in Neural Information Processing Systems*, Colorado, United States, 27 Nov –2 Dec 2000, pp. 1057–1063.

[37] N. Heess, G. Wayne, D. Silver, T. Lillicrap, T. Erez, and Y. Tassa, "Learning continuous control policies by stochastic value gradients," in *Proceedings of Advances in Neural Information Processing Systems*, Montreal, Canada, 7–12 Dec 2015, pp. 2944–2952.

[38] D. Silver, G. Lever, N. Heess, T. Degris, D. Wierstra, and M. Riedmiller, "Deterministic policy gradient algorithms," 2014.

[39] F. A. Miranda, "Advanced communications technologies in support of nasa mission," in *The European Conference on Antennas and Propagation (EuCAP 2018)*, no. GRC-E-DAA-TN53779, 2018.

[40] J. Hamkins, L. Deutsch, D. Divsalar, S. Dolinar, D. Lee, F. Stocklin, J. Wesdock, and C. Patel, "Formulation of forward error correction coding recommendations for future nasa space communications," in *2008 IEEE Aerospace Conference*. IEEE, 2008, pp. 1–18.

[41] H. Hemmati, *Deep space optical communications*. John Wiley & Sons, 2006.

[42] "Ns-3 model library," https://www.nsnam.org/docs/release/3.36/models/singlehtml/index.html, May 2022, (Accessed on June, 2022).

[43] X. He, K. Zhao, and X. Chu, "Automl: A survey of the state-of-the-art," *Knowledge-Based Systems*, vol. 212, p. 106622, 2021.

[44] A. Nagabandi, I. Clavera, S. Liu, R. S. Fearing, P. Abbeel, S. Levine, and C. Finn, "Learning to adapt in dynamic, real-world environments through meta-reinforcement learning," *arXiv preprint arXiv:1803.11347*, 2018.

[45] A. Barreto, W. Dabney, R. Munos, J. J. Hunt, T. Schaul, H. P. van Hasselt, and D. Silver, "Successor features for transfer in reinforcement learning," *Advances in neural information processing systems*, vol. 30, 2017.

[46] S. Endres, M. Griffith, B. Malakooti, K. Bhasin, and A. Holtz, "Space based internet network emulation for deep space mission applications," in *22nd AIAA International Communications Satellite Systems Conference & Exhibit 2004 (ICSSC)*, 2004, p. 3210.

[47] R. Netravali, A. Sivaraman, S. Das, A. Goyal, K. Winstein, J. Mickens, and H. Balakrishnan, "Mahimahi: Accurate record-and-replay for {HTTP}," in *Proc. USENIX Annual Technical Conference (USENIX/ATC)*, 2015, pp. 417–429.

Arooj Masood received the B.S. and M.S. degrees in computer science from the Lahore College for Women University, Lahore, Pakistan, in 2011 and 2013, respectively. She is currently pursuing the Ph.D. degree with Chung-Ang University, Seoul, South Korea. She was a Lecturer with the Department of Computer Science, Government College for Women Gulberg, Lahore. Her current research interests include edge computing, ubiquitous computing, edge caching, heterogeneous aerial access networks, LEO satellite networks, deep space networks, TCP congestion control protocols, deep reinforcement learning methods and federated learning algorithms.

Taeyun Ha received the B.S. degree in Computer Science and Engineering from Chung-Ang University, South Korea, in 2020. He is currently pursuing an M.S. degree in Computer Science and Engineering at Chung-Ang University, South Korea. His research interests include deep space networking, wireless networks, and the Internet of Things.

Demeke Shumeye Lakew received the B.S. degree in computer science from Hawassa University, Hawassa, Ethiopia in 2006 and the M.S. degree in computer science from Addis Ababa University, Addis Ababa, Ethiopia in 2011. He was a Lecturer with the College of Informatics, Kombolcha Institute of Technology (KIoT), Wollo University, Dessie, Ethiopia. He is currently pursuing the Ph.D. degree in computer science and engineering with the School of Computer Science and Engineering, Chung-Ang University, Seoul, South Korea. His current research interests include wireless communication, mobile edge computing, reinforcement learning, Internet of Things, and flying ad hoc network.

Nhu-Ngoc Dao (Senior Member, IEEE) received the B.S. degree in electronics and telecommunications from the Posts and Telecommunications Institute of Technology, Hanoi, Vietnam, in 2009, and the M.S. and Ph.D. degrees in computer science from the School of Computer Science and Engineering, Chung-Ang University, Seoul, South Korea, in 2016 and 2019, respectively. He is currently an Assistant Professor with the Department of Computer Science and Engineering, Sejong University, Seoul, South Korea. Prior to joining Sejong University, he was a Visiting Researcher with the University of Newcastle, Callaghan, NSW, Australia, in 2019 and a Postdoc Researcher with the Institute of Computer Science, University of Bern, Switzerland, from 2019 to 2020. His research interests include network softwarization, mobile cloudization, intelligent systems, and the Intelligence of Things. He is currently the Editor of the *Scientific Reports*.

Sungrae Cho is a professor with the school of computer sciences and engineering, Chung-Ang University (CAU), Seoul. Prior to joining CAU, he was an assistant professor with the department of computer sciences, Georgia Southern University, Statesboro, GA, USA, from 2003 to 2006, and a senior member of technical staff with the Samsung Advanced Institute of Technology (SAIT), Kiheung, South Korea, in 2003. From 1994 to 1996, he was a research staff member with electronics and telecommunications research institute (ETRI), Daejeon, South Korea. From 2012 to 2013, he held a visiting professorship with the national institute of standards and technology (NIST), Gaithersburg, MD, USA. He received the B.S. and M.S. degrees in electronics engineering from Korea University, Seoul, South Korea, in 1992 and 1994, respectively, and the Ph.D. degree in electrical and computer engineering from the Georgia Institute of Technology, Atlanta, GA, USA, in 2002. His current research interests include wireless networking, ubiquitous computing, and ICT convergence. He has been a subject editor of IET Electronics Letter since 2018, and was an area editor of Ad Hoc Networks Journal (Elsevier) from 2012 to 2017. He has served numerous international conferences as an organizing committee chair, such as IEEE SECON, ICOIN, ICTC, ICUFN, TridentCom, and the IEEE MASS, and as a program committee member, such as IEEE ICC, GLOBECOM, VTC, MobiApps, SENSORNETS, and WINSYS.