Multimedia Tools and Applications (2026) 85:84
https://doi.org/10.1007/s11042-026-21294-7

Adaptive scheduling for multimedia and text traffic in ocean networks through queuing theory integrated reinforcement learning

Simi Surendran[1] · Maneesha Vinodhini Ramesh[2] · Usha Kumari P V[3] · Alberto Montresor[4]

Received: 9 December 2024 / Revised: 24 July 2025 / Accepted: 23 September 2025

Abstract
The Offshore Communication Network (OCN) is an Internet of Vehicles system that connects fishing vessels at sea, enabling communication and internet access to support sustainable growth in the blue economy. The dynamic maritime environment and limited infrastructure make it challenging to ensure low-latency and high-reliability transmission, especially for multimedia messages with strict quality demands. To address these challenges, this paper proposes an adaptive transmission queue scheduling framework for multimedia and text messages in the OCN. The framework combines a multidimensional quasi-birth-and-death queuing model to analyze queue states and loss rates with Deep Reinforcement Learning (DRL) to dynamically adjust queue priorities in response to real-time network conditions. In addition to enhancing lower layer performance, the proposed solution is designed to maintain application-level quality of service for critical offshore operations, including emergency response coordination, real-time multimedia exchange for maritime safety, and mission-critical reporting. Experimental results demonstrate that the intelligent transmission queue scheduling scheme reduces queue delay and loss rate, thus improving operational efficiency and enabling IoT-driven sustainable marine resource management.

Keywords Maritime IoT network · Internet of vehicles · Deep-sea fishing communication · Adaptive transmission scheduling · Queuing theory · Reinforcement learning

1 Introduction

A significant challenge encountered in deep-sea fishing relates to the lack of affordable offshore communication solutions capable of operating over long distances from the shore. Conventional communication technologies, such as cellular networks and marine radio systems, offer limited offshore connectivity, typically extending only up to 20 km. However,

Extended author information available on the last page of the article

fishing operations often extend well beyond distances of 100 km from the shore. Currently, fishermen rely on handheld radios for communication, which are proving to be unreliable, particularly under adverse sea conditions. In addition, the coverage of the cellular network is restricted to just 15 km from the shore. Although satellite telephone systems provide communication capabilities across vast distances, their high cost makes them impractical for widespread adoption. Consequently, due to the lack of low-cost and real-time communication options, fishermen remain disconnected from the world, even in life-threatening emergencies. Real-time communication is essential for the transmission of multimedia data that supports emergency management and sustainable fishing practices. To address these limitations, Rao et al. proposed *offshore communication network* (OCN), a network of fishing vessels designed to overcome the limitations of traditional communication methods in remote maritime environments [1]. Through OCN, fishermen can utilize smartphones equipped with internet access on board for multimedia data sharing and text messaging, enabling seamless connectivity with the external world directly from their vessels.

Unlike typical vehicular networks, which are designed for land-based vehicles operating within road infrastructure, OCN must address the unique challenges presented by maritime environments. These challenges include the effects of wave-induced mobility on fishing vessels, the influence of severe weather conditions on wireless signals, the limitations in installing additional infrastructure, and the misalignment of directional antennas. Even after a network has been established, nodes may encounter unpredictable movements induced by sea waves, resulting in rapid changes in topology due to antenna orientation, vessel rocking, and propagation effects. This results in abrupt fluctuations in link quality. Yao et al. conducted a comprehensive review of the challenges in maritime networks and highlighted its similarities with terrestrial networks [2]. These factors complicate the implementation of OCN, which requires adaptations to maintain service quality.

1.1 Architecture of OCN

The *offshore communication network*, as described by Rao et al. is a heterogeneous network composed of fishing vessels that aim to provide internet connectivity across the ocean [1]. The architecture of the OCN involves a decentralized self-organizing system in which each fishing vessel operates as a node within the network. The ad-hoc nature of the network allows vessels to autonomously join or leave the system as they navigate the ocean, ensuring uninterrupted connectivity regardless of their location.

Figure 1 presents the architecture. A comprehensive description of the communication architecture and OCN routing schemes can be found in companion papers [1, 3–6]. In OCN, fishing vessels are classified into three categories based on their communication resources: access nodes, adaptive nodes, and supernodes. *Access nodes* consist solely of a wireless *access router* (AR), *adaptive nodes* have one *adaptive back-haul equipment* (ABE) and *super nodes* contains two ABEs and one AR. Adaptive and supernode is also referred to as *long-range* (LR). In fishing vessels, each AR is equipped with an omnidirectional antenna, enabling it to provide a Wi-Fi signal within a 500 m radius and connect devices like smartphones, tablets, and nearby ARs. ABEs, on the other hand, use 120° sector antennas to establish connectivity over distances of up to 20 km via long-range Wi-Fi links.

The architecture of the OCN is organized into three tiers: *Layer 0* which consists of a mesh network of access nodes utilizing Wi-Fi links; *Layer 1* which establishes the ad-hoc

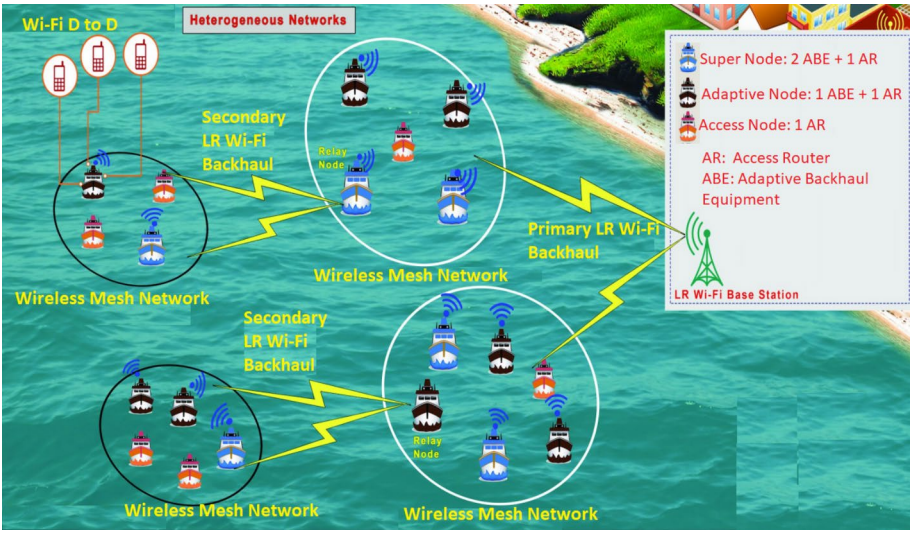

Fig. 1 Architecture of offshore communication network [1]

backbone network comprising long-range Wi-Fi nodes; *Layer 2* which constitutes the network of base stations. Machine learning techniques are employed on offline data within Layer 2 nodes, with the resulting models subsequently transmitted to edge nodes within Layers 0 and 1. These edge nodes then update the offline model parameters using real-time data. A preliminary implementation of the OCN architecture was assessed in the Arabian Sea from a coastal village in the state of Kerala, India. During field tests, the network exhibited a minimum range of 50 km in the initial hop and maintained a range of 20 km in subsequent hops.

1.2 Our contributions

In OCN, there are four types of traffic: emergency alerts, data, voice, and video, each with varying priorities. Emergency alerts require timely delivery, while real-time multimedia messages must comply with specific quality standards. The packet scheduling algorithm at the MAC layer determines the transmission sequence from the queue, ensuring efficient processing to minimize packet drops and maintain timely service delivery. In static priority systems, packets with higher priorities induce delays for lower priority packets. In contrast, dynamic prioritization methods require additional input from applications to effectively assign traffic priorities. Service differentiation in multiqueue systems requires the selection of the optimal service queue and the assignment of incoming messages to the appropriate queue to maintain the desired quality of service. Given the dynamic nature of traffic patterns, variable node density, and intermittent connectivity in OCN, an intelligent scheduling strategy is essential for accurately modeling outbound multi-priority packet traffic.

This paper addresses the challenge of efficiently managing transmission queues within a node's allocated channel access time by proposing a multi-priority queuing architecture with adaptive prioritization specifically designed for OCN. Leveraging continuous-time Markov chain theory and the quasi-birth-and-death process, we present a queuing analytical

model for the OCN transmission queue, which estimates the queue length and loss rate for different packet types. Due to dynamic traffic conditions, it is necessary to regularly update the priority of queues. A reinforcement learning agent is used to learn and adjust queue priorities based on current traffic requirements and performance metrics. When the RL agent develops a new policy for queue priorities based on the observed environment, an event-driven mechanism triggers the queue-length estimation process. This approach enables the system to recalculate and update the estimated queue lengths and loss rates, ensuring efficient and adaptive queue management.

The primary contributions of this paper are as follows.

- Propose a multi-priority queuing architecture with adaptive message prioritization designed for OCN to optimize transmission queue management.
- Develop an event-driven queuing analytical model using the quasi-birth-death process to estimate queue lengths and loss rates for different packet types in the OCN transmission queue.
- Introduce a deep reinforcement learning strategy that dynamically learns and adjusts queue priorities based on real-time traffic demands and performance metrics.

The remainder of this paper is structured as follows. Section 2 reviews previous research on the scheduling of transmission queues in wireless nodes. Section 3 discusses the quasi-birth-and-death model for OCN transmission queue management, supplemented by a deep reinforcement learning algorithm. Section 5 presents the experimental results, followed by the concluding remarks in Section 6.

2 Related work

Transmission queue scheduling plays an important role in controlling access to the shared wireless medium, particularly in extreme environments that require reliable communication. Since the queue state directly impacts the Quality of Service (QoS), numerous queue-aware scheduling algorithms have been developed to optimize packet forwarding from the queue, with the goal of maximizing throughput, minimizing delay, and ensuring fair resource allocation [7–12]. Early studies on scheduling focused solely on the network layer [13], but later cross-layer design strategies are also considered [14, 15]. For instance, Muzakkari et al. proposed an energy-efficient and QoS-aware protocol with a duty cycle scheme that adapts to queue size and priority class, reduces delays for high-priority packets and supports time-bounded delivery [8]. A survey of medium access methods for providing QoS in ad hoc networks is presented in [16].

Priority queueing and differentiated services are the fundamental techniques applied to manage traffic [17, 18]. In priority queuing systems, multiple queues are used at the MAC layer to store packets based on preferences. When the node has the opportunity to access the channel, its scheduler selects the highest priority packets. Service differentiation is the most used method, which prioritizes flows or packets and maps to quality [19]. Priority assignment can be static—based on fixed attributes like traffic type or source—or dynamic, adapting based on factors such as hop count or packet expiry [20–24]. Hybrid schemes are also used in heterogeneous environments [25]. Traffic classes may be managed using single or

multiple queues. Single-queue systems struggle with prioritization under long queues, while multi-queue architectures offer simplicity, but face challenges in accurately ranking queue importance in dynamic environments [26]. However, the main challenge in these systems is to accurately prioritize each queue in the presence of dynamic traffic.

To better understand queuing behavior, analytical models have been proposed. Ozdemir et al. modeled service time using Markov-modulated processes in single-hop wireless settings [27], while Tickoo et al. derived the average queuing delay for IEEE 802.11 networks [28]. Ray et al. used queuing theory to model linear wireless networks [29], and Bisnik et al. extended this to multihop networks, though only with stationary nodes [30]. Other models include Deepak et al.'s discrete phase-type distribution [31] and Dey et al.'s quasi-birth-and-death process for analyzing packet waiting time [32]. The authors derived statistical characteristics such as packet waiting time probability distributions, but these results were not utilized in the transmission queue scheduling. Liu et al. presented a QoS-aware MAC protocol for wireless sensor networks using a multi-queue architecture [23], and Ben et al. proposed a queue classification and scheduling model based on fixed priority levels [26]. In this protocol, queues are assigned fixed priority and packets are classified into predetermined priority levels based on the importance of the packets received from the source node. Tickoo et al. also introduced a discrete-time G / G / 1 queue analysis for IEEE 802.11 MACs [28]. Fallahi et al. examined queue dynamics due to sleep-wake mechanisms in energy-sensitive MAC protocols [33]. However, most of these analytical models do not address adaptive transmission scheduling or multi-priority traffic handling under dynamic conditions.

Recent studies have explored reinforcement learning approaches for MAC-level scheduling and queue management. Azzino et al. introduced a RL-based scheduling mechanism to determine the optimal duration of contention-free access periods in IEEE 802.11ad to improve overall network efficiency[34]. In another study, Pratama et al. proposed a low-latency Q-learning approach for 6TiSCH networks to optimize cell allocation to reduce latency[35]. In addition, deep RL techniques have been widely applied to improve traffic scheduling and resource allocation in diverse network environments. Song et al. developed a low-power queue management system based on deep RL for IoT sensor networks, ensuring QoS [36]. Wu et al. proposed an SDN-based multilayered traffic scheduling architecture for data center networks, leveraging traffic criticality and cost metrics in a Deep Q-Network [37]. Cong et al. proposed multiagent reinforcement learning to optimize task offloading in the Internet of Vehicles under dynamic network conditions [38]. Xue et al. presented a deep reinforcement learning-based task scheduling scheme for a multiuser IoV network to reduce latency[39]. In industrial IoT scenarios, Yu et al. introduced Deep-DFS, a deterministic scheduler that guarantees bounded latency using a Deep RL framework [40]. Zhang et al. presented a deep RL-based delay-aware scheduler for 5G radio access networks [41]. For time-sensitive networking over IEEE 802.11, Kim et al. proposed a deep RL-based scheduler that learns transmission patterns under dynamic conditions [42].

Deep RL has been successfully applied to more specialized and constrained environments. Ma et al. proposed a model-free queue management framework that learns optimal packet dropping strategies to mitigate buffer congestion while preserving throughput [43]. Forero et al. employed a Deep RL algorithm based on soft actors and critics to optimize bandwidth allocation and traffic prioritization in underwater acoustic networks, addressing congestion and high-delay challenges [44]. In sensor networks, Song et al. introduced

a Deep RL model to intelligently control the actions of the head of the cluster, balancing energy usage and data freshness for stable performance [36]. Finally, Stoltidis et al. explored the application of Deep RL to improve active queue management in 5G and beyond networks, offering improved adaptability and responsiveness to dynamic traffic demands [45].

The above review highlights the significant progress made in queue management and scheduling using both analytical and learning-based approaches across diverse network environments. While traditional models provide insights into queuing dynamics, they often lack adaptability to real-time, multi-priority traffic scenarios. Recent improvements in reinforcement learning offer more adaptable solutions in dynamic network conditions. Building on these developments, this work proposes a DQN-based dynamic message priority assignment scheme integrated with queuing analytical models. This approach aims to improve the management of the transmission queue and the network performance in OCNs which are characterized by resource constraints, high variability, and the need for reliable low-latency communication.

3 Quasi birth death model for OCN transmission queue management

The Quasi Birth Death (QBD) model is a mathematical framework to analyze systems characterized by a structured sequence of states and transitions. These transitions typically involve births, where entities are added to the system, and deaths, where entities are removed. A birth-death (BD) process is a Markov process on the state space $S = \{0, 1, 2, ...\}$ where transitions are between adjacent states: from state *i* to states $i + 1$ and $i - 1$. Assuming that the arrival to the queuing system follows a Poisson process and has an exponentially distributed service time, then the queuing system represents a BD process [46].

The transitions from any state *i* to any other state *j* must traverse all intermediate states without skipping any state. QBD process is a generalization of the BD process. A continuous-time QBD process $\Omega = \{(X_k, J_k),\ k \geq 0\}$ constitutes a continuous-time Markov chain with a state space $\{(0, j); 1 \leq j \leq a\} \bigcup \{(n, j);\ n \geq 1,\ 1 \leq j \leq b\}$, where *a* and *b* represent positive integers. The state space can be partitioned into levels, with each level *l* containing *k* phases. The one-step transitions from a state in level *i*, $i \geq 1$ proceed to a state in level $i - 1$, *i*, or $i + 1$. If the transition rates are independent of levels, then the QBD process is referred to as a level-independent QBD process. Since the transitions of level independent QBD process with state space $\{(0, j); 1 \leq j \leq a\} \bigcup \{(n, j); n \geq 1,\ 1 \leq j \leq b\}$ are only to the neighboring levels or within the same level, the process has a generator matrix with block-tri-diagonal structure.

In OCN, messages are categorized into four types: emergency alerts ($\mathcal{E}$), data ($\mathcal{D}$), voice ($\mathcal{A}$), and video ($\mathcal{V}$). Emergency traffic takes precedence over all other types. The priorities of the remaining three types dynamically adapt to the communication context. Each node in the network functions as a server for packet transmission, equipped with buffer space to store packets. Whenever a node needs to transmit a message, it classifies the traffic types $\mathcal{E}$, $\mathcal{D}$, $\mathcal{A}$, and $\mathcal{V}$, respectively. The node chooses a transmission policy based on the queue length that maximizes the available resources. The preliminary design of the QBD model has been discussed in prior work by Simi et al. [47]. This study expands the existing model and incorporates a reinforcement learning based queue prioritization to manage the dynamics of OCN.

The model under consideration involves a non-preemptive priority queuing system that accommodates four message types. Each message type follows an independent Poisson process with arrival rates denoted by λ_1, λ_2, λ_3 and λ_4 respectively. All queues are equipped with individual finite capacity buffers of size θ. If a buffer reaches its capacity for any message type, incoming messages of that type are lost. The priorities of these queues are context-dependent and determined by a vector $\langle p_1, p_2, p_3, p_4 \rangle$, where p_i is normalized within the range of 0 to 1. The priority of emergency alert messages, p_1, remains the highest and is statically set. The priorities of the remaining message types are dynamically adjusted using a reinforcement learning strategy detailed in Section 4.

The service times for each type of message are exponentially distributed with parameters μ_1, μ_2, μ_3, and μ_4, and are served based on probabilities p_1, p_2, p_3, and p_4. The probability of serving the queue adheres to $p_1 + p_2 + p_3 + p_4 = 1$, $p_1 > p_2$, $p_1 > p_3$, and $p_1 > p_4$. High-priority queues are preferred over low-priority queues for service. However, the arrival of a new packet in a high-priority queue does not interrupt the transmission of a packet in a lower-priority queue that is currently being served. A single-server system is assumed, as a single node processes all requests. The server provides a non-preemptive service to the highest priority queue. Upon the completion of message transmission, the server can: (a) enter an idle state if all queues are empty, (b) select a message from a non-empty queue following FIFO order if only one queue is non-empty, or (c) choose the highest priority queue and serve packets in FIFO order if multiple queues contain packets. The non-preemptive multi-priority transmission queue model is shown in Fig. 2.

The sequel uses the following notations in the node transmission queue $\mathcal{T}_q$.

n_t: Number of emergency packets in the $\mathcal{T}_q$ at time *t*.

i_t: Number of data packets in $\mathcal{T}_q$ at time *t*.

j_t: Number of audio packets in $\mathcal{T}_q$ at time *t*.

k_t: Number of video packets in $\mathcal{T}_q$ at time *t*.

The transmission queuing system for OCN can be described by the QBD process, which is a four-dimensional irreducible continuous-time Markov chain defined by (1).

$$X(t) = \{(n_t, i_t, j_t, k_t), \quad 0 \le n_t, i_t, j_t, k_t \le \theta, \quad t \ge 0\} \tag{1}$$

The states of $X(t)$ are arranged in lexicographic order as in (2).

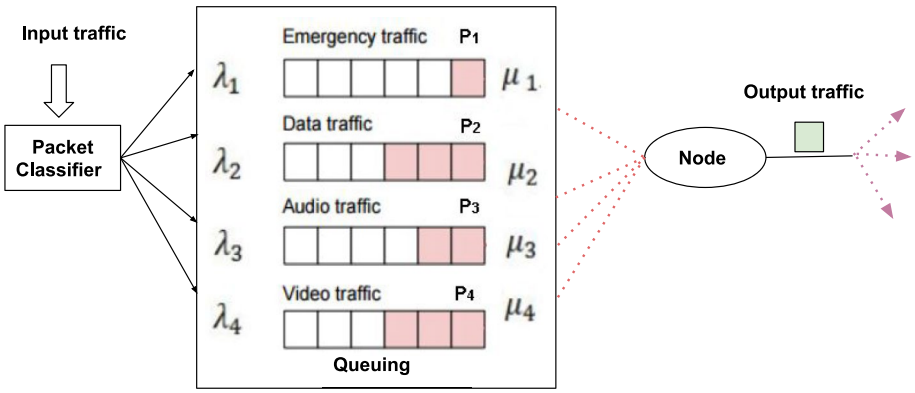

Fig. 2 Model of non-preemptive multi-priority transmission queue with single server

$$Z = \{(n_t, i_t, j_t, k_t) \mid 0 \le n_t, i_t, j_t, k_t \le \theta\} \tag{2}$$

The random variable n_t is referred to as the level of the process at time *t*, whereas the variables (i_t, j_t, k_t) represent the phase of the process at time *t*. Consequently, $\{X(t), t \ge 0\}$ is a level-independent QBD process defined on the state space $\{0, 1, ..., \theta\} \times \{0, 1, ..., \theta\} \times \{0, 1, ..., \theta\} \times \{0, 1, ..., \theta\}$ whose generator Q possesses a block tri-diagonal representation, as illustrated in matrix (3).

$$Q = \begin{bmatrix} B_0 & B_1 & 0 & 0 & 0 & \cdots & 0 \\ B_2 & A_1 & A_0 & 0 & 0 & \cdots & 0 \\ 0 & A_2 & A_1 & A_0 & 0 & \cdots & 0 \\ 0 & 0 & A_2 & A_1 & A_0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \ddots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & C_0 & C_1 \end{bmatrix} \tag{3}$$

This transition rate matrix corresponds to the definition of a QBD, where each entry in the Q matrix is a matrix, and each row sum of Q is zero. Also, for each level $n_t = 0, 1, 2, .., \theta$, $B_0 + B_1$, $B_2 + A_1 + A_0$, $A_0 + A_1 + A_2$ and $C_0 + C_1$ are equal to 0. For $n_t \ge 1$, the process can have transitions from (n_t, i_t, j_t, k_t) to $(n_t - 1, i_t, j_t, k_t)$, $(n_t + 1, i_t, j_t, k_t)$, $(n_t, i_t + 1, j_t, k_t)$, $(n_t, i_t, j_t + 1, k_t)$, $(n_t, i_t, j_t, k_t + 1)$, $(n_t, i_t - 1, j_t, k_t)$, $(n_t, i_t, j_t - 1, k_t)$ and $(n_t, i_t, j_t, k_t - 1)$. The process make transition from (n_t, i_t, j_t, k_t) to $(n_t + 1, i_t, j_t, k_t)$ with rate λ_1 and transition from (n_t, i_t, j_t, k_t) to $(n_t - 1, i_t, j_t, k_t)$ with rate μ_1 with a probability p_1. The rate of transition from (n_t, i_t, j_t, k_t) to $(n_t, i_t + 1, j_t, k_t)$, $(n_t, i_t, j_t + 1, k_t)$, $(n_t, i_t, j_t, k_t + 1)$ is λ_2, λ_3, and λ_4. Also, μ_2, μ_3, and μ_4 represents the transition rate from (n_t, i_t, j_t, k_t) to $(n_t, i_t - 1, j_t, k_t)$, $(n_t, i_t, j_t - 1, k_t)$ and $(n_t, i_t, j_t, k_t - 1)$ with probability p_2, p_3 and p_4 respectively. Then, the matrix Q has a block diagonal structure. The transition diagram is shown in Figure 3 and Figure 4. Each state of the node is represented by a 4- tuple $\langle t_1, t_2, t_3, t_4 \rangle$ where t_i is the number of messages present in $\mathcal{Q}_\mathcal{E}$, $\mathcal{Q}_\mathcal{D}$, $\mathcal{Q}_\mathcal{A}$, and $\mathcal{Q}_\mathcal{V}$. (4) to (13) shows the sub matrices of transition matrix Q.

$$A_1 = \begin{bmatrix} A_{11} & A_{12} & 0 & 0 & \cdots & 0 \\ A_{13} & A_{11} & A_{12} & 0 & \cdots & 0 \\ 0 & A_{14} & A_{11} & A_{12} & \cdots & 0 \\ 0 & 0 & A_{14} & A_{11} & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & A_{11}^* \end{bmatrix} \tag{4}$$

where $A_{11}^* = A_{11} + A_{12}$.

$$A_{11} = \begin{bmatrix} A_{111} & A_{112} & 0 & 0 & \cdots & 0 \\ A_{113} & A_{111} & A_{112} & 0 & \cdots & 0 \\ 0 & A_{113} & A_{111} & A_{112} & \cdots & 0 \\ 0 & 0 & A_{113} & A_{111} & A_{112} & 0 \\ \vdots & \vdots & \vdots & A_{113} & \ddots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & A_{111}^* \end{bmatrix} \tag{5}$$

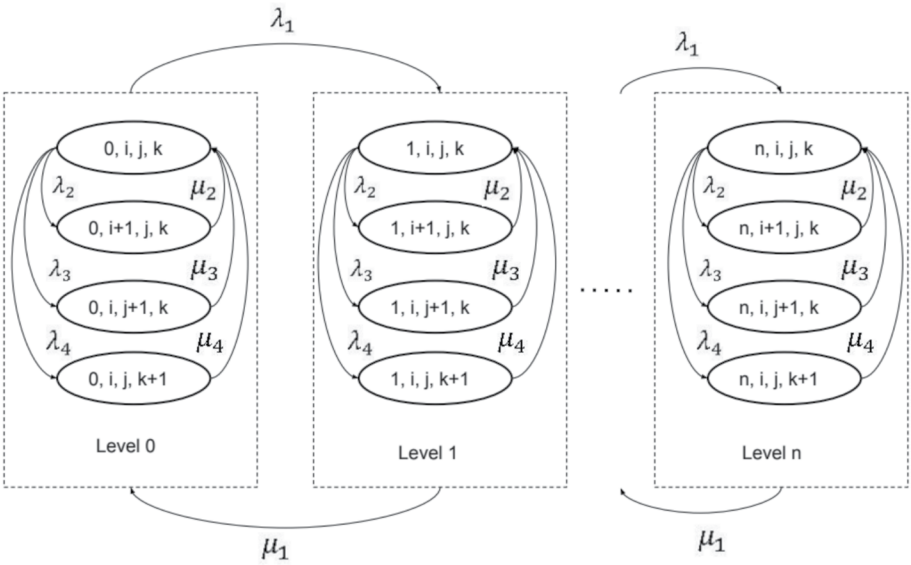

Fig. 3 Transition diagram of QBD process with four priority messages

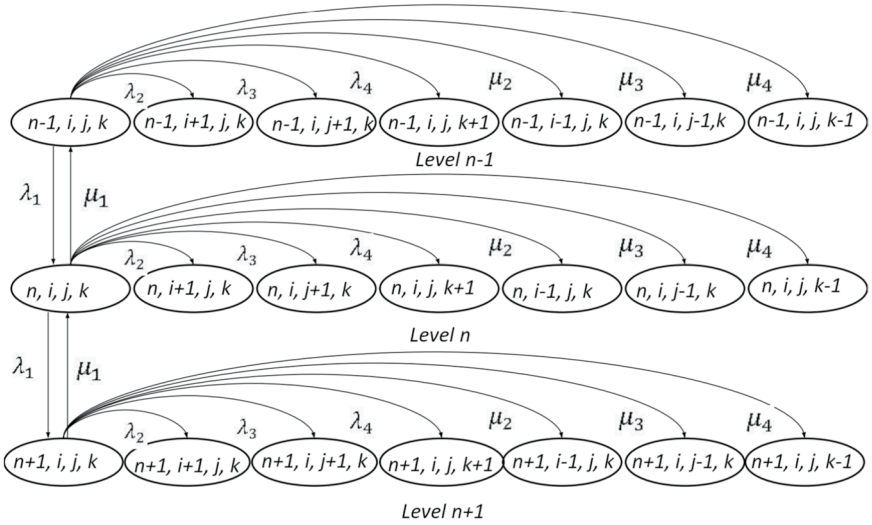

Fig. 4 Detailed transition diagram of QBD process with four priority messages

where $A^*_{111} = A_{111} + A_{112}$.

$$A_{12} = \begin{bmatrix} \lambda_2 I & 0 & \cdots & 0 \\ 0 & \lambda_2 I & 0 & 0 \\ \vdots & & \ddots & \vdots \\ 0 & 0 & \cdots & \lambda_2 I \end{bmatrix} \tag{6}$$

$$A_{13} = \begin{bmatrix} \mu_3 & 0 \\ 0 & \mu_2 I \end{bmatrix} \tag{7}$$

$$A_{14} = \begin{bmatrix} \mu_2 I & 0 & \cdots & 0 \\ 0 & \mu_2 I & 0 & 0 \\ \vdots & 0 & \ddots & \vdots \\ 0 & 0 & \cdots & \mu_2 I \end{bmatrix} \tag{8}$$

$$\mathrm{A}_{111} = \begin{pmatrix} -(\lambda_2+\lambda_3+\lambda_4+\mu_2+\mu_3) & \lambda_4 & & \\ \mu_4 & t & & \\ \vdots & \vdots & \cdots & \\ 0 & 0 & & \\ 0 & \cdots & 0 & \\ \lambda_4 & \cdots & 0 & \\ \cdots\ 0 & \cdots & 0 & \\ \vdots & \ddots & \vdots & \\ 0 & \cdots & -(\lambda_2+\lambda_3+\mu_2+\mu_3+\mu_4) & \end{pmatrix} \tag{9}$$

where $t = -(\lambda_2+\lambda_3+\lambda_4+\mu_2+\mu_3+\mu_4)$

$$A_{112} = \lambda_3 . I \tag{10}$$

$$A_{113} = \mu_3 . I \tag{11}$$

$$A_2 = \begin{bmatrix} \lambda_1 I & 0 & \cdots & 0 \\ 0 & \lambda_1 I & 0 & 0 \\ \vdots & 0 & \ddots & \vdots \\ 0 & 0 & \cdots & \lambda_1 I \end{bmatrix} \tag{12}$$

$$A_0 = \begin{bmatrix} 0 & 0 & 0 & \cdots & 0 \\ \mu_1 I & 0 & 0 & 0 & 0 \\ 0 & \mu_1 I & 0 & 0 & 0 \\ \vdots & 0 & \mu_1 I & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 0 \end{bmatrix} \tag{13}$$

3.1 Steady state analysis

For determining the stability condition, we define $A = A_0 + A_1 + A_2$ as shown in (14).

$$A = \begin{bmatrix} D_1 & L_2 & 0 & 0 & 0 & \cdots & 0 \\ M_2 & D_2 & L_2 & 0 & 0 & \cdots & 0 \\ 0 & M_2 & D_2 & L_2 & 0 & \cdots & 0 \\ 0 & 0 & M_2 & D_2 & L_2 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \ddots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & M_2 & D_3 \end{bmatrix} \tag{14}$$

where

$$L_2(j,k) = \begin{cases} \lambda_2 & \text{if } k = j \\ 0 & otherwise \end{cases}$$

$$M_2(j,k) = \begin{cases} \mu_2 p_2 & \text{if } k = j \\ 0 & otherwise \end{cases}$$

Let $\phi = \{\phi_0, \phi_1, .., \phi_{\theta-1}\}$ be the steady-state vector of A. Then ϕ satisfies the equations $\phi A = 0, \phi e = 1$.

$$[\phi_0, \phi_1, ..., \phi_{\theta-1}] \begin{bmatrix} D_1 & L_2 & 0 & 0 & 0 & \cdots & 0 \\ M_2 & D_2 & L_2 & 0 & 0 & \cdots & 0 \\ 0 & M_2 & D_2 & L_2 & 0 & \cdots & 0 \\ 0 & 0 & M_2 & D_2 & L_2 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \ddots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & M_2 & D_3 \end{bmatrix} = 0$$

$$\begin{aligned} \phi_0 D_1 + \phi_1 M_2 &= 0 \\ \phi_0 L_2 + \phi_1 D_2 + \phi_2 M_2 &= 0 \\ &\vdots \\ \phi_{\theta-2} L_2 + \phi_{\theta-1} D_3 &= 0 \end{aligned}$$

$$\phi_i = \begin{cases} \phi_{i-1} D_1 {M_2}^{-1}, & \text{if } i = 1 \\ -[\phi_{i-2} L_2 + \phi_{i-1} D_2] {M_2}^{-1}, & \text{if } i = 2, 3, .., \theta - 2 \\ -[\phi_{i-1} L_2] {D_3}^{-1}, & \text{if } i = \theta - 1 \end{cases} \tag{15}$$

We can obtain all values of ϕ in terms of ϕ_0 from (15).

3.2 Stability condition

The matrix $A = A_0 + A_1 + A_2$ is considered to examine the system stability. This is the infinitesimal generator of the finite state continuous time Markov chain $X(t) = \{(n_t, i_t, j_t, k_t), \ 0 \le n_t, i_t, j_t, k_t \le \theta, \ t \ge 0\}$. Let $\pi_l = (\pi_{0,}\pi_{1,}\pi_{2,} \ldots \pi_k)$ be the stationary probability vector in the QBD process with a generator matrix shown in (14). The i^{th} element of π_l represents the stationary probability that the QBD process is in phase *i* of level *l*.

Theorem 1 *The multi-queue system with dynamic priorities under study is stable if and only if* $\lambda_1 < \mu_1 p_1$.

Proof This Markov chain is stable if and only if the left drift rate exceeds the right drift rate.

$$\pi A_0 e \ < \ \pi A_2 e \tag{16}$$

From the matrices A_0, A_2, we have $\pi A_0 e = \lambda_1 \sum_{i=1}^{\theta} \pi_i e$ and $\pi A_2 e = \mu_1 p_1 \sum_{i=1}^{\theta} \pi_i e$. Using the relation 16, the stability condition $\lambda_1 < \mu_1 p_1$ is obtained. Thus a multi-queue system with dynamic priorities under study is stable.

3.3 Steady state probability vector

Assuming that the stability condition in (16) is satisfied, the computation of the steady state probability of the system state is defined as follows:

Let π denote the steady-state probability vector of the generator Q. Then we have

$$\pi Q = 0, \quad \pi e = 1 \tag{17}$$

Partitioning π as $\pi = (\pi_0, \pi_1, \pi_2, ...)$ then each of the sub vectors as:

$x_{n_1} = \{x_{n_1}(n_2, n_3, n_4); \quad 0 \leq n_2, n_3, n_4 \leq \theta, \quad n_1 \geq 0\}$.

π under the assumption that the stability condition in (16) holds is obtained as:

$$\pi_n = \pi_{n-1} R; \quad n \geq 1 \tag{18}$$

where R is the minimal non-negative solution of the matrix quadratic equation:

$$R^2 A_2 + R A_1 + A_0 = 0 \tag{19}$$

and the boundary equations are given by

$$\pi_0 B_0 + \pi_1 B_1 = 0, \quad \pi_0 B_1 + \pi_1 A_1 + \pi_2 A_2 = 0 \tag{20}$$

The normalizing condition (17) gives

$$\pi_0 e + \pi_1 (I - R)^{-1} e = 1 \tag{21}$$

To compute matrix R, the logarithmic reduction algorithm is used [48].

3.4 Performance measures

Let n_1, n_2, n_3, n_4 be the number of each type of message at time t in the node's transmission queue. We define the performance measures as follows:

- *Expected number of messages*: The expected number of messages in each sub queue is denoted by $\eta_1, \eta_2, \eta_3, \eta_4$ and it is formulated as

$$\eta_1 = \sum_{n_1=0}^{\infty} \sum_{n_2=0}^{\theta} \sum_{n_3=0}^{\theta} \sum_{n_4=0}^{\theta} n_1 x_{n_1}(n_2, n_3, n_4)$$

$$\eta_2 = \sum_{n_1=0}^{\infty} \sum_{n_2=0}^{\theta} \sum_{n_3=0}^{\theta} \sum_{n_4=0}^{\theta} n_2 x_{n_1}(n_2, n_3, n_4)$$

$$\eta_3 = \sum_{n_1=0}^{\infty} \sum_{n_2=0}^{\theta} \sum_{n_3=0}^{\theta} \sum_{n_4=0}^{\theta} n_3 x_{n_1}(n_2, n_3, n_4)$$

$$\eta_4 = \sum_{n_1=0}^{\infty} \sum_{n_2=0}^{\theta} \sum_{n_3=0}^{\theta} \sum_{n_4=0}^{\theta} n_4 x_{n_1}(n_2, n_3, n_4)$$

- *Loss rate*: The loss rate in each queue is given by

$$\mathcal{L}_{\mathcal{E}} = \lambda_1 \sum_{n_1=0}^{\infty} \sum_{n_2=0}^{\theta} \sum_{n_3=0}^{\theta} \sum_{n_4=0}^{\theta} x_{n_1}(n_2, n_3, n_4)$$

$$\mathcal{L}_{\mathcal{D}} = \lambda_2 \sum_{n_1=0}^{\infty} \sum_{n_3=0}^{\theta} \sum_{n_4=0}^{\theta} x_{n_1}(\theta, n_3, n_4)$$

$$\mathcal{L}_{\mathcal{A}} = \lambda_3 \sum_{n_1=0}^{\infty} \sum_{n_2=0}^{\theta} \sum_{n_4=0}^{\theta} x_{n_1}(n_2, \theta, n_4)$$

$$\mathcal{L}_{\mathcal{V}} = \lambda_4 \sum_{n_1=0}^{\infty} \sum_{n_3=0}^{\theta} \sum_{n_4=0}^{\theta} x_{n_1}(n_3, n_4, \theta)$$

- Probability of server being busy

$$\mathcal{B} = \sum_{n_1=0}^{\infty} \sum_{n_2=0}^{\theta} \sum_{n_3=0}^{\theta} \sum_{n_4=0}^{\theta} x_{n_1}(n_2, n_3, n_4)$$

To estimate queue length and loss rate using the QBD model, the system first initializes the parameters, which include arrival and service rates for each queue level. A transition matrix is subsequently constructed to define the probabilities of transitioning between different queue states. Matrix-analytic methods are applied to solve the QBD model, enabling us to analyze queue length and loss rate. In the dynamic OCN environment, it is essential to adjust queue priorities to align with real-time traffic conditions. As queue priorities change, transition probabilities must also be updated, while arrival rates are adjusted to reflect fluctuations in network traffic, and service rates are modified according to the new priorities. For this purpose,an intelligent method is required to adaptively learn optimal queue priorities. In the following subsection, we discuss how a reinforcement learning strategy can be employed to learn these priorities and effectively update queue estimates.

4 Reinforcement learning based transmission queue scheduling

A significant challenge in multiqueue architectures is determining which queue to serve next. In static priority queuing systems, packets from higher priority queues are prioritized, resulting in increased delays for lower priority packets. To address this issue, dynamic adjustment of queue priorities is essential. We introduce a deep reinforcement learning approach that adaptively modifies queue priorities based on observed traffic patterns and performance metrics within the OCN environment [49]. By integrating domain knowledge from queuing theory into DRL frameworks, the system can autonomously learn priorities that respond to fluctuating traffic demands.

Figure 5 shows the architecture of the priority-based DRL scheduler. The scheduler assigns priority to each queue except the emergency queue in each epoch and observes the performance metrics. Based on the performance, the priority assignment has to be done in the next stage. This dynamic priority assignment is modeled as a reinforcement learning problem. Each node acts as an agent, and the environment sets all queues. The agent's goal is to minimize transmission delay and loss rate from the queue.

The Markov Decision Process for modeling the DRL agent is defined as follows:

- *State*: The state of the agent at time t, denoted as S_t consists of the queue lengths of the dynamic priority queues, its current priority, and performance parameters. It is defined as $S_t = \left\{\vec{Q}_{\text{exp}}, \vec{P}, \vec{\eta}\right\}$, where $\vec{Q}_{\text{exp}} = [E[L_{\mathcal{D}}], E[L_{\mathcal{A}}], E[L_{\mathcal{V}}]]$ represents the expected queue lengths for the data, audio, and video traffic types, derived from the QBD model. The vector $\vec{P} = [P_{\mathcal{D}}, P_{\mathcal{A}}, P_{\mathcal{V}}]$ denotes the current priority assignment for each queue, where higher values indicate higher transmission priority. The performance metrics $\vec{\eta}$ include the packet loss rates $\vec{\eta}_{\text{loss}} = [\rho_{\mathcal{D}}, \rho_{\mathcal{A}}, \rho_{\mathcal{V}}]$ and the average transmission delays $\vec{\eta}_{\text{delay}} = [d_{\mathcal{D}}, d_{\mathcal{A}}, d_{\mathcal{V}}]$, computed over a recent time window. Also, $\vec{\eta}$ includes the estimated traffic arrival rates $\vec{\lambda} = [\lambda_{\mathcal{D}}, \lambda_{\mathcal{A}}, \lambda_{\mathcal{V}}]$, and the link utilization factor u, which gives the proportion of available bandwidth currently in use.

- *Action*: Agent action is the priority assignment for each queue. $A = a_1, a_2, ..., a_m$ where a_i corresponds to the priority level of queue i, where each $a_i \in \{1, 2, \ldots, P\}$.

$$A = \{(a_1, a_2, \ldots, a_Q) \mid a_i \in \{1, 2, \ldots, P\}, \forall i \in [1, Q]\}$$

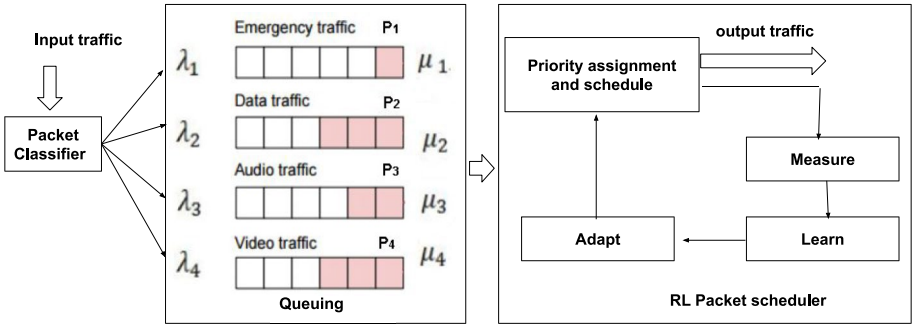

Fig. 5 Architecture of RL-based packet scheduler

To avoid assigning the same priority to multiple queues, the agent selects from the set of all possible permutations of distinct priorities assigned to Q queues. The total number of possible actions in this case is $Q!$. This allows the DQN agent to choose a valid priority configuration.

- *Reward function*: The reward function is formulated so that a node receives a reward inversely proportional to the length of the queue and the loss rate, as shown in (22). At each time step t, the agent receives a reward R_t. For each queue $\mathcal{Q}_i$, the performance metrics include the throughput T_i, the packet arrival rate λ_i, the packet loss rate L_i, and the average transmission delay d_i. The reward function is defined as:

$$R_t = \sum_{i=1}^{N} \omega_i \left(\alpha \cdot \frac{T_i}{\lambda_i} - \beta \cdot \ L_i - \gamma \cdot d_i \right) \tag{22}$$

where ω_i is the weight assigned to queue $\mathcal{Q}_i$. The parameters α, β, and γ control the emphasis on improving throughput and penalizing packet loss and delay, thereby balancing the quality of service requirements.
To solve the aforementioned MDP, we employ a Deep Q-Network (DQN) algorithm as in Algorithm 1, which assists the agent in learning an optimal priority assignment. At each epoch, the agent observes the current state, which includes queue lengths, current priority assignments, and performance metrics. Based on this state, the agent selects an action, defined as a priority configuration of all queues. The DQN approximates $Q(s, a)$ using a deep neural network. Experience replay is used to store agent-environment interactions in a buffer, and then randomly sampling from this buffer to train the neural network. A target network is maintained to compute stable Q-value targets and is periodically updated with weights from the primary network. An epsilon-greedy exploration strategy is used, where the agent selects a random action with probability ϵ and the action with the highest Q-value with probability $1 - \epsilon$. To enhance the convergence speed of the DQN, we use domain knowledge derived from the analytical queuing model to initialize the Q-values associated with each state-action pair. This initialization enables the agent to start with better values rather than relying on random exploration. The model generates estimates of performance indicators, including expected queue lengths and packet loss rates under specific traffic arrival patterns and service configurations. Based on this analysis, we implement the following initialization scheme:

- If any queue in a given state exceeds the threshold for both queue length and packet loss rate, the corresponding priority configuration is initialized with a higher Q-value Q_h. This prioritizes the actions that solve congestion or reduce loss.
- If none of the queues exceed the thresholds, the Q-value for that state-action pair is set to a lower value Q_l.

The implementation of the DRL agent adopts an approach suitable for resource constrained environments, such as fishing vessels, where real-time training is impractical due to limited computational resources. To address this computational issue, the DRL model is initially trained offline using historical data that reflect traffic patterns and varying environmental

conditions. This training is conducted in high performance systems located in an onshore control center. Once the training phase is completed, the model is deployed onboard the vessel, where it operates in inference mode. In this setup, the agent performs real-time queue scheduling decisions with minimal computational overhead. To maintain adaptability in dynamic network conditions, the system collects performance feedback during operation. This feedback is periodically transmitted to the onshore control center. Based on these data, the DRL model is re-trained and updated versions of the model are sent back to the vessel for redeployment.

Algorithm 1 DQN for dynamic queue prioritization.

Input: State space $\mathcal{S}$, Action space $\mathcal{A}$, empirical QBD model
Initialize:
Initialize Q-network with random weights θ
Initialize target network with weights $\theta^- \leftarrow \theta$
Initialize experience replay buffer $\mathcal{D}$
Initialize Q-table $Q(s, a)$ using empirical estimates from QBD model:
foreach *state* $s \in \mathcal{S}$ **do**
  foreach *action* $a \in \mathcal{A}$ **do**
    Use QBD model to estimate queue length and loss under (s, a)
    if *any queue exceeds threshold* **then**
      $Q(s, a) \leftarrow Q_h$
    end
    else
      $Q(s, a) \leftarrow Q_l$
    end
  end
end
for *episode = 1 to M* **do**
  Initialize environment and observe initial state s_0 **for** *t = 1 to T* **do**
    a. With probability ϵ, select random action $a_t \in \mathcal{A}$
    Otherwise, select $a_t = \arg\max_a Q(s_t, a; \theta)$
    b. Execute action a_t, observe reward r_t, and next state s_{t+1}
    c. Store (s_t, a_t, r_t, s_{t+1}) in replay buffer $\mathcal{D}$
    d. Sample minibatch from $\mathcal{D}$
    foreach *sample* (s_j, a_j, r_j, s_{j+1}) **do**
      i. Compute target: $y_j = r_j + \gamma \cdot \max_{a'} Q(s_{j+1}, a'; \theta^-)$
      ii. Calculate loss and perform gradient descent
      $\mathcal{L}(\theta) = \left(y_j - Q(s_j, a_j; \theta)\right)^2$
    end
    e. Update target network
    if $t \mod C = 0$ **then**
      $\theta^- \leftarrow \theta$
    end
    f. Perform periodic update using QBD estimates
    $s_t \leftarrow s_{t+1}$
  end
end
return θ

5 Performance evaluation

To assess the effectiveness of the proposed intelligent transmission queue scheduling scheme, we developed a simulation environment that models OCN. The simulation was implemented in Python and integrates a dynamic wireless communication framework that leverages both Wi-Fi and long-range Wi-Fi technologies. The network model, including link behavior and traffic prediction parameters, is adapted from our earlier work presented in [50]. Each node of the vessel in the simulation experiences packet arrivals modeled as a Poisson process, capturing the characteristics of continuous traffic such as text, image, and multimedia transmissions. The service process is represented using a multi-priority queuing system with variable service rates to reflect realistic maritime communication dynamics. The proposed scheduling algorithm was evaluated by analyzing queue size and packet loss rate under various combinations of packet arrival and service rates. The QBD solver was utilized to estimate queue lengths and loss rates, which were subsequently integrated into the agent's learning process. Whenever an update in queue priorities occurs, the agent triggers an event, prompting the QBD solver to update its estimates. These two components interact in a continuous loop, collaboratively optimizing queue management.

The evaluation of the DRL-based dynamic priority queuing model considers several factors, including expected message rates, loss rates, and server utilization. In addition, the performance of the proposed model is compared with two distinct scheduling schemes in a four-queue system. Model 1 implements a first-come-first-served (FCFS) scheduling scheme, whereas Model 2 utilizes priority queuing with static priorities assigned to each queue. Model 3 represents the proposed dynamic priority queuing approach, in which the priority of the emergency queue is assigned statically, whereas the priorities of the remaining queues are dynamically determined using the reinforcement learning methodology.

The variation in the arrival rate of each message type is examined and the expected queue lengths are observed under different models. Figure 6 illustrates the expected queue length and server utilization rate with an increase in the arrival rate of data messages while keeping all other arrival and service rates constant. As the arrival rate λ_2 of the data messages increases, Model 3 tries to adapt the priority of the data queue, and the lowest expected queue length is noted for $\mathcal{Q}_{\mathcal{D}}$ in Fig. 6a. Since Model 1 gives priority to the first arrived packets and Model 2 uses fixed priority, the expected queue length of $\mathcal{Q}_{\mathcal{D}}$ is higher when the arrival rate of the data messages increases. Due to the dynamic change in the packet arrival rate of the data message, Model 2 can not serve $\mathcal{Q}_{\mathcal{D}}$ at a higher rate. However, DRL based Model 3 learns the behavior of the traffic and adapts the priorities. The expected lengths of $\mathcal{Q}_{\mathcal{A}}$ and $\mathcal{Q}_{\mathcal{V}}$ is shown in Fig. 6b and c. For $\mathcal{Q}_{\mathcal{A}}$, the expected length in Model 3 is higher than Model 1 since it adjusts the priority of $\mathcal{Q}_{\mathcal{D}}$ and $\mathcal{Q}_{\mathcal{A}}$ to provide better service for $\mathcal{Q}_{\mathcal{D}}$. In this scenario, Model 3 gives the lowest priority to $\mathcal{Q}_{\mathcal{V}}$, and hence the highest queue length for $\mathcal{Q}_{\mathcal{V}}$. Figure 6d shows the server busy time in each model with an increase in λ_2 and observes that server utilization is also maximum in Model 3.

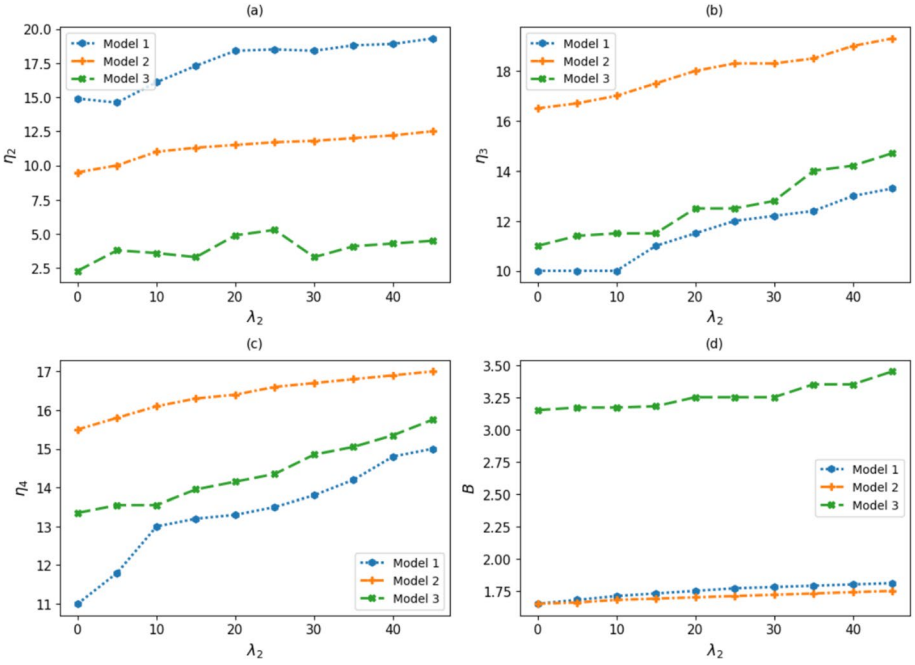

Fig. 6 Comparison of expected queue length and server busy rate with increase in the arrival rate of data messages (λ_2): (**a**) Expected length of messages in data queue (η_2) is the lowest for Model 3 (**b**) Expected length of messages in audio queue (η_3) in Model 3 is in between Model 1 and Model 2 (**c**) Expected length of messages in video queue (η_4) is the highest in Model 3 (**d**) server busy rate is the highest in Model 3.

With an increase in λ_2, the loss rate in each queue is plotted as shown in Fig. 7. The loss rate in $\mathcal{Q}_\mathcal{D}$ is higher for both Model 1 and Model 2, but Model 3 reduces the loss rate by adjusting the service rate. In $\mathcal{Q}_\mathcal{A}$ as well, Model 3 demonstrates superior performance relative to static priority assignment. The loss rate in $\mathcal{Q}_\mathcal{V}$ is also comparable to that of Model 2. In addition, the influence of the arrival rate of audio and video messages is examined. Figure 8 illustrates the fluctuations in each queue. In this context, Model 3 assigns greater weights to $\mathcal{Q}_\mathcal{A}$, resulting in the expected queue lengths in $\mathcal{Q}_\mathcal{D}$ and $\mathcal{Q}_\mathcal{A}$ being minimized within Model 3. Consequently, the priority of $\mathcal{Q}_\mathcal{V}$ diminishes, leading to a decline in performance in Model 3 in $\mathcal{Q}_\mathcal{V}$. Increased server utilization is observed in Model 3 with an increase in λ_3.

The loss rate with an increase in λ_3 is plotted in Fig. 9. As the arrival rate of audio messages increases, $\mathcal{Q}_\mathcal{A}$ experiences the highest loss; however, the loss rate in Model 3 is lower compared to the other two models. Moreover, in $\mathcal{Q}_\mathcal{D}$ and $\mathcal{Q}_\mathcal{E}$, Model 3 also demonstrates better performance. Due to the adjustment of traffic priorities, the loss rate in $\mathcal{Q}_\mathcal{V}$ is higher in Model 3 compared to Model 1.

Figure 10 shows the status of queue length with an increase in the arrival rate of video messages. When λ_4 increases, $\mathcal{Q}_\mathcal{D}$ and $\mathcal{Q}_\mathcal{A}$ obtain average performance, while $\mathcal{Q}_\mathcal{V}$ receives the better performance in Model 3.

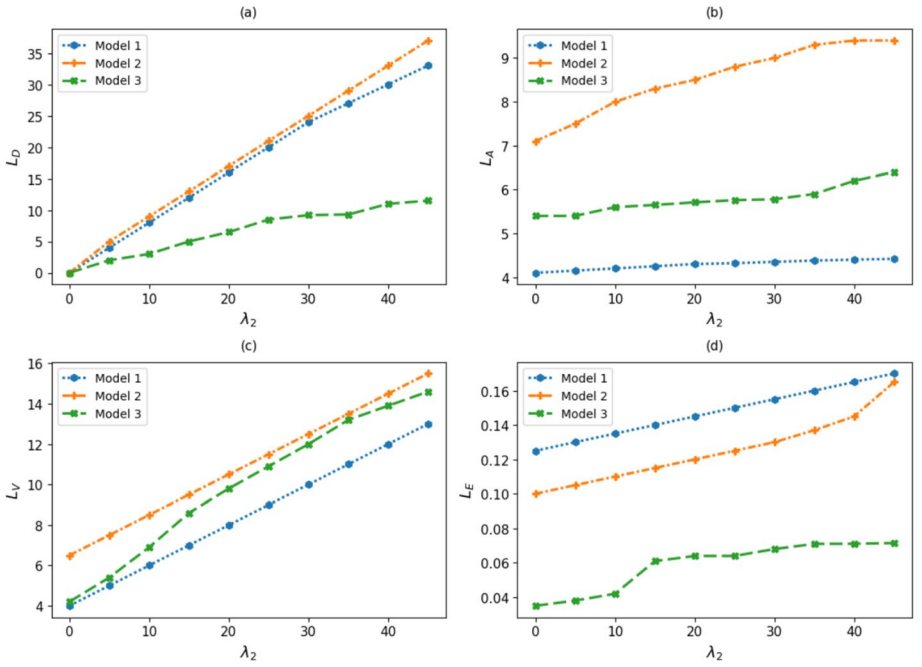

Fig. 7 Comparison of loss rate rate with increase in arrival rate of data messages (λ_2): (**a**) Loss rate of $\mathcal{Q}_\mathcal{D}$ (**b**) Loss rate of $\mathcal{Q}_\mathcal{A}$ (**c**) Loss rate of $\mathcal{Q}_\mathcal{V}$ (**d**) Loss rate of $\mathcal{Q}_\mathcal{E}$

Since the highest priority is assigned statically to $\mathcal{Q}_\mathcal{E}$, the expected number of messages and loss rate is negligible for $\mathcal{Q}_\mathcal{E}$.

5.1 Discussion

The results of this study demonstrate the effectiveness of integrating queueing models with DRL to optimize transmission queue scheduling in OCN. The proposed dynamic scheduling scheme was evaluated under simulated conditions with dynamic traffic loads and varying packet arrival rates. The QBD model was used to analyze the expected number of messages in the queue and the loss rate in four types of traffic: emergency, audio, video, and data. The DRL-based agent learns to dynamically assign transmission priorities to each queue in response to real-time traffic fluctuations, thus improving QoS.

The performance of this DRL model (Model 3) was compared against two conventional scheduling schemes:FCFS and static priority queueing. The comparative analysis reveals that Model 3 consistently achieves better performance in terms of queue lengths and packet loss rates. As the arrival rates of different message types increase, Model 3 adapts its prioritization policy in real time, avoiding performance degradation across lower-priority queues. In contrast, static models exhibit limitations in handling fluctuating traffic, often resulting in excessive queue buildup and higher loss rates. The DRL-based approach was

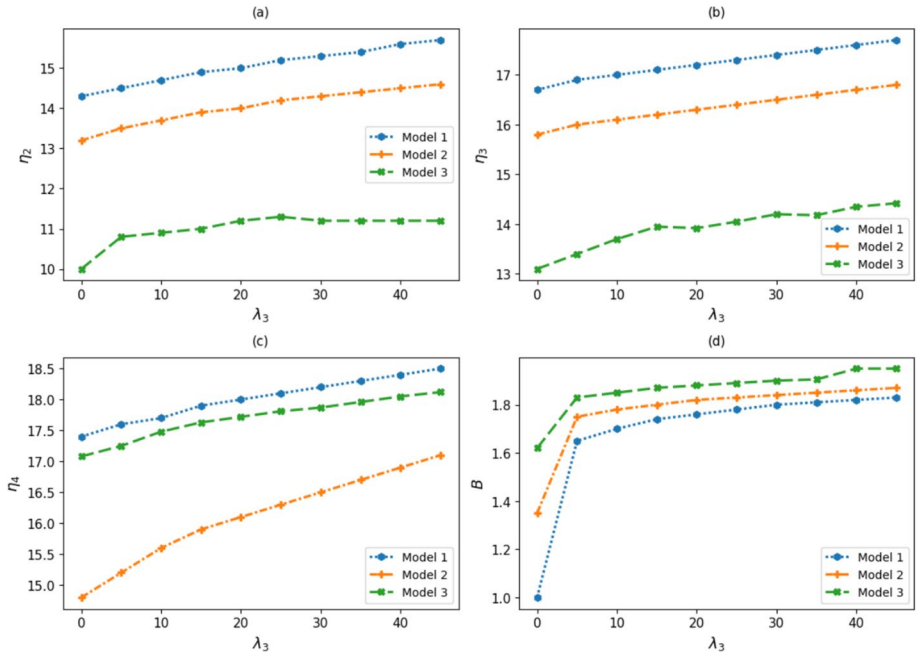

Fig. 8 Comparison of expected queue length and server busy rate with increase in the arrival rate of audio messages(λ_3): (**a**) Expected length of $\mathcal{Q}_{\mathcal{D}}$ is lowest for model 3 (**b**) Expected length of $\mathcal{Q}_{\mathcal{A}}$ in Model 3 is lowest (**c**) Expected length of $\mathcal{Q}_{\mathcal{V}}$ is in Model 3 is close to Model 1 and greater than Model 2 (**d**) Server busy rate is the highest in Model 3

observed to maintain balanced performance by dynamically prioritizing queues based on current traffic loads. Model 3 also demonstrated a better packet loss management, particularly for high priority traffic, while maintaining acceptable loss rates for lower priority message types. This adaptive behavior ensures that QoS communication is preserved even under dynamic traffic.

Although the integration of QBD-based analytical modeling with DRL-driven adaptive scheduling presents an effective solution for queue management in OCNs, some practical limitations arise when deploying the system on resource constrained platforms such as fishing vessels. One major challenge is the limited computational capacity on board, making real-time training of deep reinforcement learning models infeasible. This issue is addressed by conducting offline training at an onshore control center, with only the model deployed onboard to perform real-time queue scheduling. However, the system still depends on intermittent connectivity to the shore for periodic model updates. Another practical limitation involves the onboard traffic classification process. Variability in signal quality and environmental interference can affect the reliability of traffic categorization, which is an important factor for the agent to make scheduling decisions. These challenges can be addressed through the use of lightweight

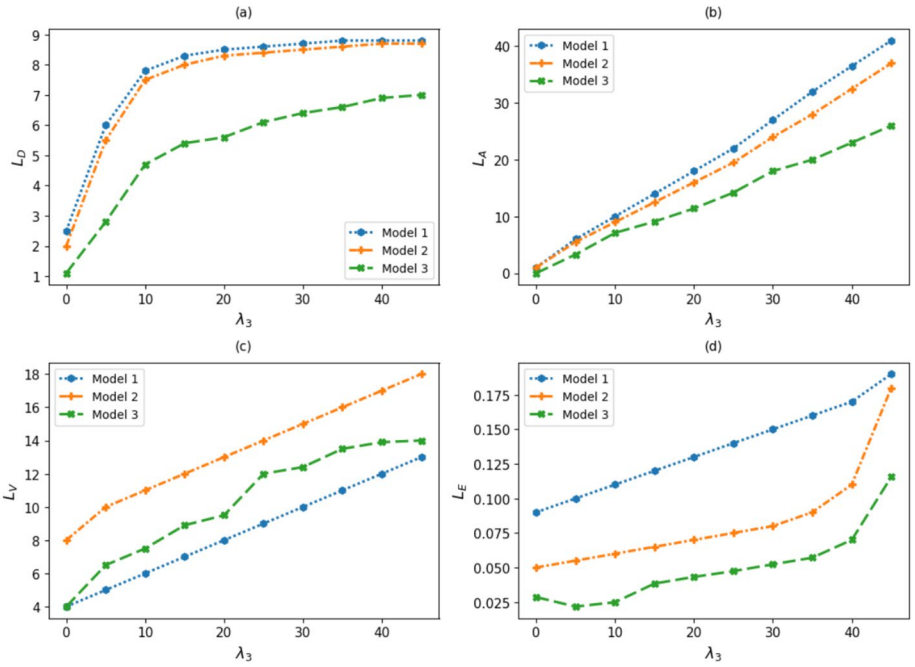

Fig. 9 Comparison of loss rate with increase in arrival rate of audio messages-λ_3: (**a**) Loss rate of $\mathcal{Q}_{\mathcal{D}}$ (**b**) Loss rate of $\mathcal{Q}_{\mathcal{A}}$ (**c**) Loss rate of $\mathcal{Q}_{\mathcal{V}}$ (**d**) Loss rate of $\mathcal{Q}_{\mathcal{E}}$

models and edge-optimized strategies specifically designed for low-power maritime communication systems.

6 Conclusion

Management of the transmission queue is essential to minimize packet loss and ensure high quality service in networks that handle heterogeneous multimedia and text traffic. This paper addressed this challenge in the context of OCNs, where limited channel access time and dynamic maritime conditions complicate scheduling decisions. We proposed a multi-priority queuing architecture with adaptive prioritization to optimize queue management within each node's transmission opportunity. By combining queuing theory with deep reinforcement learning, the framework introduced an event-driven queuing analytical model based on the quasi-birth-and-death process to estimate queue lengths and loss rates for various message types. A deep reinforcement learning agent continuously adapts queue priorities in response to real-time traffic demands. When new policies are learned, the system recalculates the queue statistics, ensuring timely and efficient prioritization. The simulation results demonstrate that the proposed approach

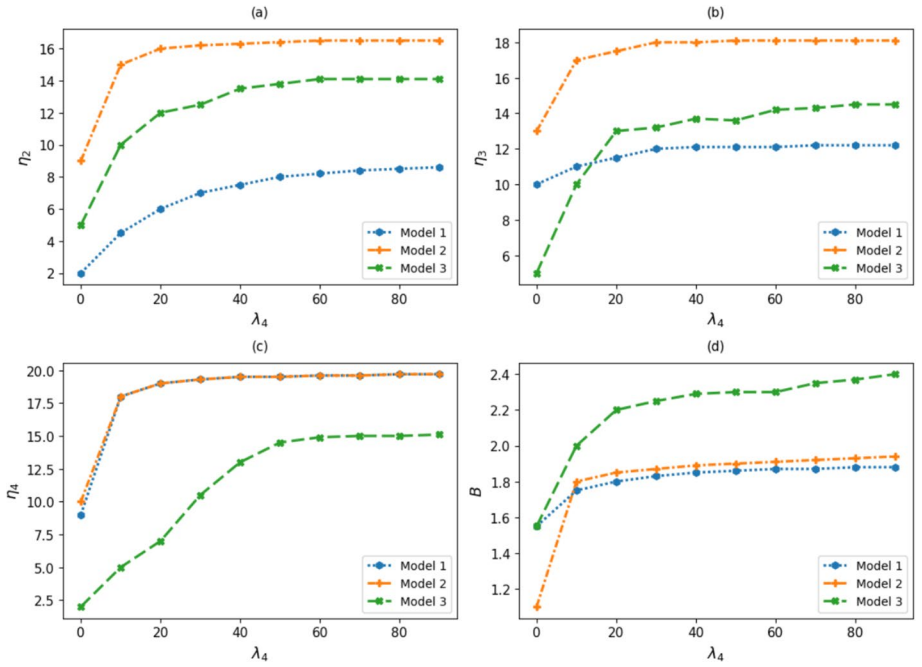

Fig. 10 Comparison of expected queue length and server busy rate with increase in the arrival rate of video messages (λ_4): (**a**) Expected length of $\mathcal{Q}_{\mathcal{D}}$ is lowest for Model 1 (**b**) Expected length of $\mathcal{Q}_{\mathcal{A}}$ in Model 1 is the lowest (**c**) Expected length of $\mathcal{Q}_{\mathcal{V}}$ is the lowest in Model 3 (**d**) server busy rate is the highest in Model 3

reduced transmission delays and the loss rate. Beyond improving lower-layer performance, this research supports higher-layer quality of service to enable reliable multimedia communication.

Author Contributions All authors contributed to the design, implementation, and evaluation of the work, as well as to the preparation of the manuscript.

Funding We wish to confirm that there has been no significant financial support for this work that could have influenced its outcome.

Data Availability This study uses simulation-based data, which can be reproduced using the parameters described in the manuscript.

Declarations

Conflict of Interest On behalf of all authors, the corresponding author states that there is no conflict of interest.

Ethics approval and consent to participate Not applicable.

Consent for publication All authors confirm that consent for publication has been obtained for any data, images, or content requiring it.

Materials availability Not applicable.

Competing Interests Not applicable.

References

1. Rao SN, Ramesh MV, Rangan V (2016) Mobile infrastructure for coastal region offshore communications and networks. In: Proc. of the IEEE global humanitarian technology conference (GHTC'16), pp99–104. https://doi.org/10.1109/GHTC.2016.7857266. IEEE
2. Yau K-LA, Syed AR, Hashim W, Qadir J, Wu C, Hassan N (2019) Maritime networking: Bringing internet to the sea. IEEE Access 7:48236–48255. https://doi.org/109/ACCESS.2019.2909921
3. Rao S.N, Raj D, Aiswarya S, Unni S(2016) Realizing cost-effective marine internet for fishermen. In: 14th Int. Symposium on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks (WiOpt'16), pp1–5. https://doi.org/10.1109/WIOPT.2016.7492904. IEEE
4. Unni S, Raj D, Sasidhar K, Rao S (2015) Performance measurement and analysis of long range wi-fi network for over-the-sea communication. In: Proc. of the 13th Int.symposium on modeling and optimization in mobile, Ad Hoc, and Wireless Networks (WiOpt'15), pp36–41. https://doi.org/10.1109/WIOPT.2015.7151030. IEEE
5. Surendran S, Ramesh MV, Montresor A (2021) Predictive analytics integrated multi-level optimization of offshore connectivity in ocean network. In: 2021 IEEE 46th Conference on Local Computer Networks (LCN), pp621–628. https://doi.org/10.1109/LCN52139.2021.9525021. IEEE
6. Dhivvya JP, Rao SN, Simi S (2017) Towards maximizing throughput and coverage of a novel heterogeneous maritime communication network. In: Proc. of the 18th ACM Int. symposium on mobile ad hoc networking and computing, pp 39. https://doi.org/10.1145/3084041.3084077. ACM
7. Stüber T, Osswald L, Lindner S, Menth M (2023) A survey of scheduling algorithms for the time-aware shaper in time-sensitive networking (tsn). IEEE Access. https://doi.org/10.1109/ACCESS.2023.3286370
8. Muzakkari BA, Mohamed MA, Kadir MF, Mamat M (2020) Queue and priority-aware adaptive duty cycle scheme for energy efficient wireless sensor networks. IEEE Access 8:17231–17242. https://doi.org/10.1109/ACCESS.2020.2968121
9. Nosheen S, Khan JY (2021) Quality of service-and fairness-aware resource allocation techniques for ieee802. 11ac wlan. IEEE Access 9:25579–25593. https://doi.org/10.1109/ACCESS.2021.3051983
10. Ahmed N, Hussain MI (2023) A qos-aware scheduling with node grouping for ieee 802.11 ah. Wireless Netw 29(4):1799–1814. https://doi.org/10.1007/s11276-022-03206-3
11. Li J, Han T, Guan W (2024) Lian X A preemptive-resume priority mac protocol for efficient bsm transmission in uav-assisted vanets. Appl Sci 14(5):2151. https://doi.org/10.3390/app14052151
12. Sarang S, Stojanović GM, Drieberg M, Stankovski S, Bingi K (2023) Jeoti V Machine learning prediction based adaptive duty cycle mac protocol for solar energy harvesting wireless sensor networks. IEEE Access 11:17536–17554. https://doi.org/10.1109/ACCESS.2023.3246108
13. Tassiulas L (1993) Ephremides A Dynamic server allocation to parallel queues with randomly varying connectivity. IEEE Trans Inf Theory 39(2):466–478. https://doi.org/10.1109/18.212277
14. Mohanram C (2007) Bhashyam S Joint subcarrier and power allocation in channel-aware queue-aware scheduling for multiuser ofdm. IEEE Trans Wireless Commun 6(9):3208–3213. https://doi.org/10.1109/TWC.2007.06030103
15. Song G, Li Y, Cimini LJ (2009) Joint channel-and queue-aware scheduling for multiuser diversity in wireless ofdma networks. IEEE Trans Commun 57(7):2109–2121. https://doi.org/10.1109/TCOMM.2009.07.070394
16. Natkaniec M, Kosek-Szott K, Szott S (2012) Bianchi G A survey of medium access mechanisms for providing qos in ad-hoc networks. IEEE communications surveys & tutorials 15(2):592–620. https://doi.org/10.1109/SURV.2012.060912.00004
17. Zhao J, Guo Z, Zhang Q, Zhu W (2002) Performance study of mac for service differentiation in ieee 802.11. In: Proc. of Global Telecommunications Conference, vol 1, pp778–782. https://doi.org/10.1109/GLOCOM.2002.1188184. IEEE
18. Yu T (2006) Lin K-J Qcws: an implementation of qos-capable multimedia web services. Multimed Tools Appl 30:165–187. https://doi.org/10.1007/s11042-006-0020-8

19. Bhatnagar S, Deb B, Nath B (2001) Service differentiation in sensor networks. In: Proc. of wireless personal multimedia communications. Citeseer
20. Saxena N, Roy A (2008) Shin J Dynamic duty cycle and adaptive contention window based qos-mac protocol for wireless multimedia sensor networks. Comput Netw 52(13):2532–2542. https://doi.org/10.1016/j.comnet.2008.05.009
21. Liu Z, Elhanany I (2006) Rl-mac: A qos-aware reinforcement learning based mac protocol for wireless sensor networks. In: Proc. of International Conference on Networking, Sensing and Control, pp 768–773. https://doi.org/10.1109/ICNSC.2006.1673243. IEEE
22. Nguyen K, Nguyen T, Chaing C.K, Motani M A prioritized mac protocol for multihop, event-driven wireless sensor networks. In: Proc. of First Int. conference on communications and electronics, pp47–52 (2006). https://doi.org/10.1109/CCE.2006.350836. IEEE
23. Liu Y, Elhanany I, Qi H (2005) An energy-efficient qos-aware media access control protocol for wireless sensor networks. In: Proc. of Int. conference on mobile adhoc and sensor systems conference, pp 3. https://doi.org/10.1109/MAHSS.2005.1542798. IEEE
24. Caccamo M, Zhang LY, Sha L, Buttazzo G (2002) An implicit prioritized access protocol for wireless sensor networks. In: Proc. of 23rd Real-Time Systems Symposium, pp39–48. https://doi.org/10.1109/REAL.2002.1181560. IEEE
25. Yigitel MA, Durmaz Incel O, Ersoy C (2010) Diff-mac: a qos-aware mac protocol with differentiated services and hybrid prioritization for wireless multimedia sensor networks. In: Proc. of the 6th ACM workshop on QoS and security for wireless and mobile networks, pp 62–69
26. Ben-Othman J, Mokdad L, Yahya B (2011) An energy efficient priority-based qos mac protocol for wireless sensor networks. In: Proc. of international conference on communications, pp1–6. https://doi.org/10.1109/icc.2011.5962414. IEEE
27. Özdemir M, McDonald AB (2004) An m/mmgi/1/k queuing model for ieee 802.11 ad hoc networks. In: Proceedings of the 1st ACM international workshop on performance evaluation of wireless Ad Hoc, Sensor, and Ubiquitous Networks, pp107–111. https://doi.org/10.1145/1023756.1023776
28. Tickoo O, Sikdar B (2004) A queueing model for finite load ieee 802.11 random access mac. In: 2004 IEEE international conference on communications (IEEE Cat. No. 04CH37577), vol 1, pp175–179. https://doi.org/10.1109/ICC.2004.1312475. IEEE
29. Ray S, Starobinski D, Carruthers JB (2005) Performance of wireless networks with hidden nodes: A queuing-theoretic analysis. Comput Commun 28(10):1179–1192. https://doi.org/10.1016/j.comcom.2004.07.024
30. Bisnik N, Abouzeid AA (2009) Queuing network models for delay analysis of multihop wireless ad hoc networks. Ad Hoc Networks 7(1):79–97. https://doi.org/10.1145/1143549.1143704
31. Deepak T (2017) A queuing network model for delay and throughput analysis in multi-hop wireless ad hoc networks. Reliability: Theory & Applications 12(2 (45))
32. Dey S (2019) Deepak T A matrix analytic approach to study the queuing characteristics of nodes in a wireless network. Opsearch 56(2):477–496. https://doi.org/10.1007/s12597-019-00373-4
33. Fallahi A, Hossain E, Alfa AS (2006) Qos and energy trade off in distributed energy-limited mesh/relay networks: A queuing analysis. IEEE Trans Parallel Distrib Syst 17(6):576–592. https://doi.org/10.1109/TPDS.2006.76
34. Azzino T, Ropitault T, Zorzi M (2020) Scheduling the data transmission interval in ieee 802.11 ad: A reinforcement learning approach. In: 2020 International conference on computing, networking and communications (ICNC), pp602–607. https://doi.org/10.1109/ICNC47757.2020.9049674. IEEE
35. Pratama YH, Chung S-H, Fawwaz DZ (2024) Low-latency and q-learning-based distributed scheduling function for dynamic 6tisch networks. IEEE Access. https://doi.org/10.1109/ACCESS.2024.3384869
36. Song T, Kyung Y (2024) Deep reinforcement learning based age-of-information-aware low-power active queue management for iot sensor networks. IEEE Internet Things J. https://doi.org/10.1109/JIOT.2024.3355410
37. Wu G (2024) Deep reinforcement learning based multi-layered traffic scheduling scheme in data center networks. Wireless Netw 30(5):4133–4144. https://doi.org/10.1007/s11276-021-02883-w
38. Cong Y, Liu M, Wang C, Sun S, Hu F, Liu Z, Wang C (2024) Task scheduling and power allocation in multiuser multiserver vehicular networks by noma and deep reinforcement learning. IEEE Internet Things J 11(13):23532–23543. https://doi.org/10.1109/TVT.2024.3427814
39. Xue J, Yu K, Zhang T, Zhou H, Zhao L, Shen X (2024) Cooperative deep reinforcement learning enabled power allocation for packet duplication urllc in multi-connectivity vehicular networks. IEEE Trans Mob Comput 23(8):8143–8157. https://doi.org/10.1109/JIOT.2024.3387072
40. Yu H, Taleb T, Zhang J (2022) Deep reinforcement learning-based deterministic routing and scheduling for mixed-criticality flows. IEEE Trans Industr Inf 19(8):8806–8816. https://doi.org/10.1109/TII.2022.3222314
41. Zhang T, Shen S, Mao S, Chang G-K (2020) Delay-aware cellular traffic scheduling with deep reinforcement learning. In: GLOBECOM 2020-2020 IEEE global communications conference, pp 1–6. https://doi.org/10.1109/GLOBECOM42002.2020.9322560

42. Kim H, Kim Y-J, Kim W-T (2024) Deep reinforcement learning-based adaptive scheduling for wireless time-sensitive networking. Sensors 24(16):5281. https://doi.org/10.3390/s24165281
43. Ma H, Xu D, Dai Y, Dong Q (2021) An intelligent scheme for congestion control: When active queue management meets deep reinforcement learning. Comput Netw 200:108515. https://doi.org/10.1016/j.comnet.2021.108515
44. Forero PA, Zhang P, Radosevic D (2021) Active queue-management policies for undersea networking via deep reinforcement learning. In: OCEANS 2021: San Diego–Porto, pp 1–8. https://doi.org/10.23919/OCEANS44145.2021.9706025
45. Stoltidis A, Choumas K, Korakis T (2025) Active queue management in 5g and beyond cellular networks using machine learning. Comput Commun 236:108108. https://doi.org/10.1016/j.comcom.2025.108108
46. Neuts MF (1981) Matrix-geometric solutions in stochastic models, volume 2 of Johns Hopkins Series in the Mathematical Sciences. Johns Hopkins University Press, Baltimore, Md
47. Simi S, Ramesh MV, Ushakumari P (2020) Queue size estimation of nodes in a heterogeneous ocean network with multiple priority traffic. Int J Veh Inf Commun Syst 5(1):26–40. https://doi.org/10.1504/IJVICS.2020.107180
48. Stewart WJ (2009) Probability, Markov Chains, Queues, and Simulation: the Mathematical Basis of Performance Modeling. Princeton University Press
49. Sutton RS, Barto AG (1999) Reinforcement learning: An introduction. Robotica 17(2):229–235. https://doi.org/10.1017/S0263574799271172
50. Surendran S, Ramesh MV, Montresor A, Montag MJ (2023) Link characterization and edge-centric predictive modeling in an ocean network. IEEE Access 11:5031–5046. https://doi.org/10.1109/ACCESS.2023.3235387

Authors and Affiliations

Simi Surendran[1] · Maneesha Vinodhini Ramesh[2] · Usha Kumari P V[3] · Alberto Montresor[4]

✉ Simi Surendran
simisurendran@am.amrita.edu

Maneesha Vinodhini Ramesh
maneesha@amrita.edu

Usha Kumari P V
ushakumari@am.amrita.edu

Alberto Montresor
alberto.montresor@unitn.it

[1] Department of Computer Science and Engineering, Amrita School of Computing, Amrita Vishwa Vidyapeetham, Amritapuri, India

[2] Amrita Center for Wireless Networks & Applications (AmritaWNA), Amrita School of Engineering, Amrita Vishwa Vidyapeetham, Amritapuri, India

[3] Department of Mathematics, Amrita School of Arts and Science, Amrita Vishwa Vidyapeetham, Amritapuri, India

[4] Department of Information Engineering and Computer Science, University of Trento, Trento, Italy