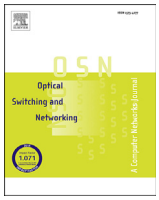
Optical Switching and Networking 37 (2020) 100567

Contents lists available at ScienceDirect

Optical Switching and Networking

journal homepage: www.elsevier.com/locate/osn



Towards self-adaptive bandwidth allocation for low-latency communications with reinforcement learning

Lihua Ruan *, Maluge Pubuduni Imali Dias, Elaine Wong

Dept. Electrical and Electronics Engineering, The University of Melbourne, Melbourne, VIC, 3010, Australia

ARTICLE INFO

Keywords:
Tactile Internet
Low-latency communication
Reinforcement learning
Resource allocation
Optical access networks

ABSTRACT

Emerging applications such as remotely-controlled human-to-machine and tactile-haptic applications in the Internet evolution demand stringent low-latency transmission. In realising these applications, current communication networks need to reduce their latency towards a millisecond order. In our previous study, we exploited supervised learning-based machine learning techniques in analysing and optimising bandwidth allocation decisions in access networks to achieve low latency. In this paper, we propose a reinforcement learning-based solution to facilitate adaptive bandwidth allocation in access networks, without needing supervised training and prior knowledge of the underlying networks. In our proposed scheme, the central office estimates the rewards of different bandwidth decisions based on the network latency resulting from executing these decisions. The reward estimates are then used to select decisions that reduce the latency in turn. In particular, we discuss the algorithms that can be used to estimate the rewards and achieve decision selection in the proposed scheme. With extensive simulations, we analyse the performance of these algorithms in diverse network scenarios and validate the effectiveness of the proposed scheme in reducing network latency over existing schemes.

1. Introduction

The Internet-of-Things and Tactile Internet evolution envisions a plethora of low-latency applications such as remotely-controlled human-to-machine applications (H2M), haptic teleoperation and real-time virtual and augmented reality applications in the next-generation communication networks [1]. These applications demand ultra-low latency in their transmission. For example, stringent low latency in 1–10 ms is required for H2M and haptic communication [2,3]. As such, current communication networks need to lower their latency for emerging applications. This relies on upgrading the underlying infrastructure in conjunction with effective network control and resource allocation strategies [4]. Existing literature, along with our previous studies, has considered converged wireless and optical access networks (PONs) with edge cloudlets in supporting low-latency applications such as shown in Fig. 1 [5,6]. Such a converged architecture benefits from both the mobility and coverage of wireless access and the high reliability and capacity provided by PONs. By strategically placing edge servers and cloudlets at the central office (CO) and/or the ONUs that integrate wireless front-ends, services can be brought close to end users and the latency can be reduced [7–9]. Nonetheless, when delivering applications converged over PONs, the latency bottleneck lies in the uplink transmission since the ONUs contend uplink bandwidth as shown in Fig. 1. Particularly, in meeting the 1–10 ms latency requirement of emerging H2M and tactile-haptic applications, the latency in the PON segment can be only up to a few hundred microseconds[4]. As such, bandwidth allocation solutions that reduce the latency caused by the contentions among ONUs are warranted.

Specifically, uplink bandwidth allocation to ONUs is mainly achieved by implementing dynamic bandwidth allocation (DBA) schemes. Typical DBA schemes rely on a report-and-grant process, whereby an ONU requests bandwidth in a report and the CO allocates bandwidth accordingly via responding a gate message [10,11]. Depending on how the CO determines the bandwidth value to be allocated, existing DBA schemes can be categorised as classic schemes and predictive schemes [12]. In classic DBA schemes such as the fixed-service and limited-service DBA, the CO primarily allocates bandwidth equal to that requested by the ONUs. In comparison, the predictive DBA schemes harness traffic prediction to estimate an additional amount of bandwidth, denoted as BW_{pred}, to the requested amount. This surplus bandwidth BW_{pred} can be a static value, such as prescribed by the constant and linear credit DBA schemes [10,13]. The BW_{pred} can also be dynamic

* Corresponding author.
E-mail address: ruanl@student.unimelb.edu.au (L. Ruan).

https://doi.org/10.1016/j.osn.2020.100567
Received 11 October 2019; Received in revised form 5 February 2020; Accepted 24 February 2020
Available online 29 February 2020

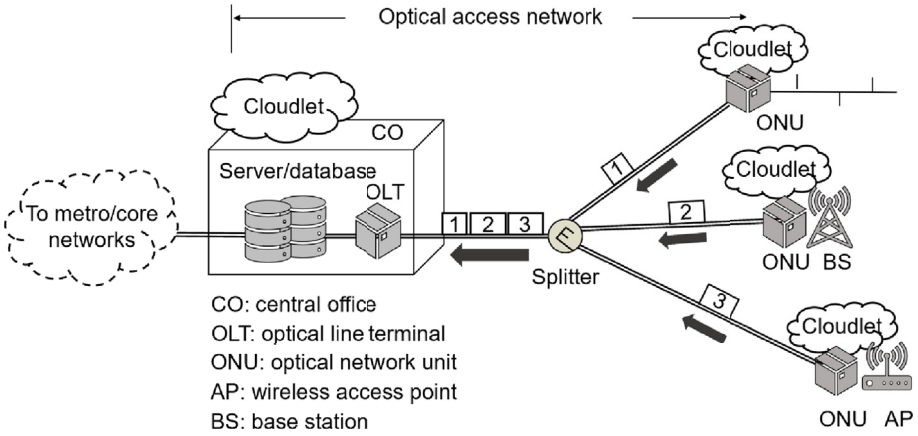


Fig. 1. Converged low-latency application delivery over access networks.

based on statistical estimation of incoming arrivals to ONUs. The predictive schemes that incorporate such estimation are known as statistical predictive DBA schemes [14]. Based on these facts, the BW_{pred} is an important bandwidth decision in existing DBA schemes that impacts the uplink latency performance. In predictive schemes, prior knowledge, primarily the statistics of arrivals, is required to yield a BW_{pred}. The BW_{pred} values are determined using specific estimation algorithms, such as the Bayesian estimation used in Ref. [15] and moving average in Ref. [16,17].

In our previous study in Ref. [18], we exploited an artificial neural network (ANN) to comprehensively compare and analyse the latency performance of existing DBA schemes. This analysis is achieved by first training an ANN to characterise the association between the latency and BW_{pred}. Then, using the trained ANN, the latency of existing DBA schemes that execute different BW_{pred} in allocating bandwidth is presented and analysed. By optimising BW_{pred} using the trained ANN, we show that existing DBA schemes, including both the classic and predictive schemes, are not optimal in terms of reducing latency. Harnessing the trained ANN, the impact of BW_{pred} allocation in DBA schemes on the latency is characterised. However, in terms of DBA implementation, a supervised training phase is required to train an ANN. Optimising BW_{pred} for bandwidth allocation needs to be performed in an offline manner. Moreover, using the trained ANN also requires prior knowledge of the underlying network such as configurations and traffic loads [18,19]. These prerequisites prevent the CO from promptly determining suitable BW_{pred} in an unknown or changing network environment. In light of the above, we investigate novel bandwidth allocation schemes that can optimise this BW_{pred} in reducing latency without relying on supervised training or knowledge of the network environment. Such solutions will be beneficial in improving existing algorithm-specific bandwidth allocation schemes towards an autonomous and intelligent fashion.

To this end, this paper presents a reinforcement learning (RL)-based bandwidth allocation scheme. RL is a special branch in machine learning (ML) that focuses on the interactions between an agent and the environment [20]. In RL, the agent reinforces its actions for more rewarding outcomes by strategically exploring and exploiting actions. It is known that a scheme that explores an optimal policy over time would be superior than the ones that adopt fixed policies [21]. As such, RL can be a promising technique in innovating bandwidth allocation schemes in PONs [22], and to the best of our knowledge, this potentiality has not been fully explored. In this paper, we formulate the bandwidth allocation in PONs as a multi-armed bandit RL problem. The CO (as the agent) adaptively adjusts bandwidth decision BW_{pred} (actions) based on the latency experienced by the packets (rewards). In RL, the CO estimates rewards by trying different actions. We present different RL algorithms that can be used to achieve reward estimation and bandwidth decision selection in our proposed scheme. The performance of these algorithms are then compared and analysed. The effectiveness of the proposed scheme in reducing the latency compared to existing schemes is validated with extensive simulations.

The rest of the paper is organised as follows. Section II introduces the existing DBA schemes and our proposed RL-based scheme. In Section III, we present solution algorithms in making bandwidth decisions in the proposed scheme. Section IV presents the simulation performance analysis. Finally, conclusions arising from this work are presented in Section V.

2. Problem formulation

2.1. Principles of DBA schemes

Fig. 2 presents the round-robin bandwidth allocation by the CO in existing DBA schemes. Two ONUs are presented for illustrative purposes. The R and G in the figure represent a report and gate message, respectively. As introduced earlier, ONUs request bandwidth using a report and the CO grants bandwidth via responding a gate. The time interval between consecutive transmissions of an ONU is known as a polling cycle. During each polling cycle, the CO determines the bandwidth to be allocated based on that requested, BW_{req}, by ONUs in their reports. Then, gate messages are sent to ONUs to indicate the uplink transmission start time and duration. Let us denote the average uplink latency as D_{uplink}, referring to the waiting time upon packets arriving at ONUs until they are transmitted to the CO.

The fixed-service DBA scheme is a simple scheme, by which the CO allocates a fixed amount of bandwidth to each ONU in each polling cycle. The limited-service DBA scheme is a commonly-adopted baseline scheme. In this scheme, the CO allocates $\min\{BW_{req}, BW_{max}\}$, where BW_{max} is the maximum bandwidth that can be allocated to an ONU. As such, compared to the fixed-service scheme, the limited-service DBA achieves low latency while preventing channel monopolisation by heavily-loaded ONUs. The predictive DBA schemes allocate an additional BW_{pred} to BW_{req} as shown in Fig. 2. The bandwidth allocated to each ONU by the CO during each polling cycle is therefore $\min\{BW_{req} + BW_{pred}, BW_{max}\}$. Allocating BW_{pred} in predictive schemes allows arriving packets at ONUs to be transmitted without needing to be reported, thereby reducing latency. Note that the limited-service DBA scheme is a special case of predictive schemes when $BW_{pred} = 0$ is constantly allocated. Based on the above technical details, we consider BW_{pred} as a vital parameter to be determined by the CO.

In the predictive schemes, BW_{pred} is primarily estimated based on traffic features such as network load or arrival rate at ONUs. For example, a static BW_{pred} is allocated for constant bit rate (CBR) arrivals to ONUs, i.e., constant credit DBA scheme [10]. By estimating packet

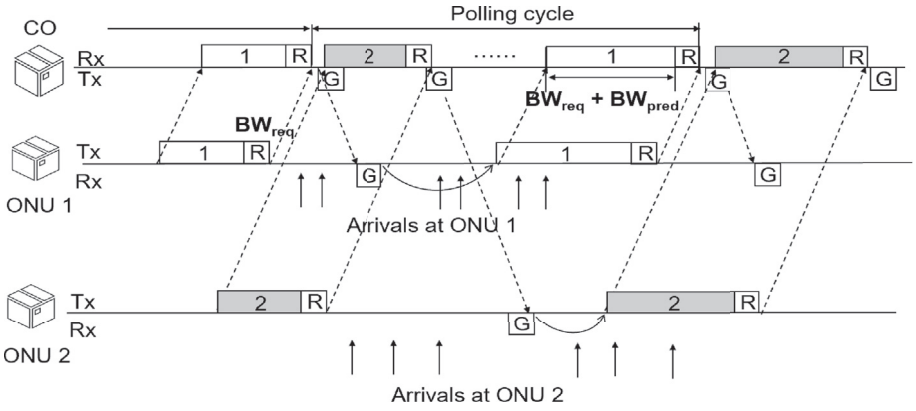


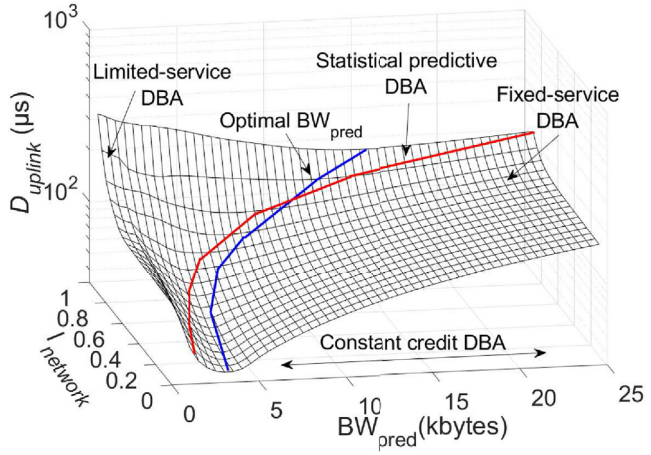
Fig. 2. An illustration of DBA operation timing diagram.

inter-arrival time or arrival rate to ONUs such as in the statistical predictive DBA schemes [15,23], BW_{pred} can be estimated using $BW_{pred} = \lambda \times T_{poll} \times L_{packet}$, where λ is the estimated arrival rate, T_{poll} is the duration of a polling cycle, and L_{packet} is the average packet length. It can be noted that the above BW_{pred} estimation requires knowledge of the feature of network traffic, e.g., CBR, an estimation of the arrival rate λ, etc. To analyse the latency performance of existing DBA schemes, we exploited supervised learning-based ML techniques in our previous study [18]. The impact of BW_{pred} on D_{uplink} was analysed using an artificial neural network (ANN). The details of our analysis and main conclusions are as follows.

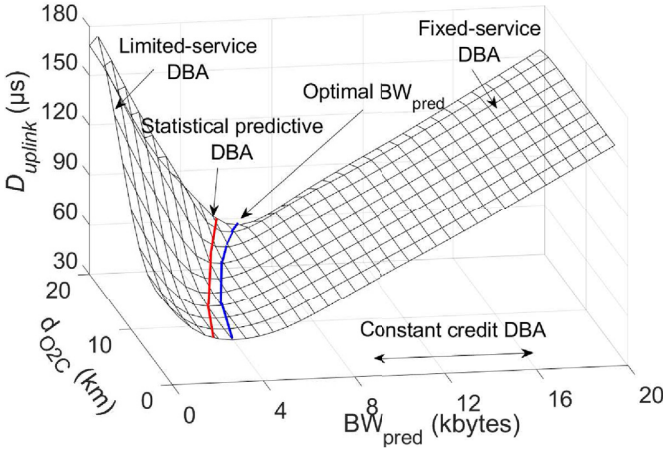
2.2. Supervised learning-based bandwidth allocation decision

In this section, we present the understandings on exiting DBA schemes attained by using an ANN. An ANN adopts layered learning architecture, i.e., input, output, and hidden layers, comprising neuron units, which are non-linear activation functions to map inputs to an output [24]. With supervised training, an ANN iteratively adjusts the parameters in its neuron units and the weights and bias in each layer to yield a desired output. As such, ANNs are advantageous in characterising complex associations among input and output features. Existing schemes estimate BW_{pred} mainly based on the arrival rate, equivalently the traffic load, and using specific algorithms such as discussed above. In our study, we take multiple network features including the traffic load, ONU-to-CO distance, the number of ONUs and packet statistics into account and train an ANN in making BW_{pred} decisions for these different network scenarios.

Via supervised training, we train a multi-layered ANN to learn the association between D_{uplink} and BW_{pred} together with the multiple network features discussed above. When the training is complete, given any BW_{pred} and features of the underlying network, the trained ANN can yield an output, which is the average latency D_{uplink} resulting from executing this BW_{pred} in DBA operation. Detailed supervised training process is explained in Ref. [18]. In Fig. 3, we show the analysis on the performance of existing DBA schemes using the trained ANN. Fig. 3(a) illustrates an example of the D_{uplink} corresponding to different BW_{pred} in a 10 Gbps PON comprising 16 ONUs and 10 km ONU-to-CO fiber links. Analytical results show that the BW_{pred} estimated and allocated in existing DBA schemes is not optimal in reducing D_{uplink}. Either bandwidth under-granting such as $BW_{pred} = 0$ in the limited-service scheme or over-granting such as in the fixed-service scheme indicated in Fig. 3 yields a high D_{uplink}. In comparison, based on the network traffic load, denoted by $l_{network}$ in Fig. 3(a), the statistical predictive DBA scheme estimates a BW_{pred} along the red curve as plotted in the figure. Fig. 3(b) further shows the BW_{pred} and D_{uplink} in PONs with different ONU-to-CO distances, denoted by d_{O2C}. This result implicates that multiple network features, not only the traffic load, impact the selection of BW_{pred} in



(a)BW_{pred} vs $l_{network}$



(b)BW_{pred} vs d_{O2C}

Fig. 3. Latency analysis using the trained ANN as a function of BW_{pred} and network features such as $l_{network}$ and d_{O2C} [18].

reducing D_{uplink}. Compared to the BW_{pred} in existing schemes, the optimal BW_{pred} values that minimise D_{uplink} corresponding to $l_{network}$ and d_{O2C} are highlighted in the blue curve in Fig. 3.

From the above analysis, in both existing schemes and the use of ANN for DBA, prior knowledge of the underlying network such as network configuration and traffic load is needed. Compared to existing schemes, the use of ANN yields better BW_{pred} decisions that reduce latency since multiple network features are considered in determining

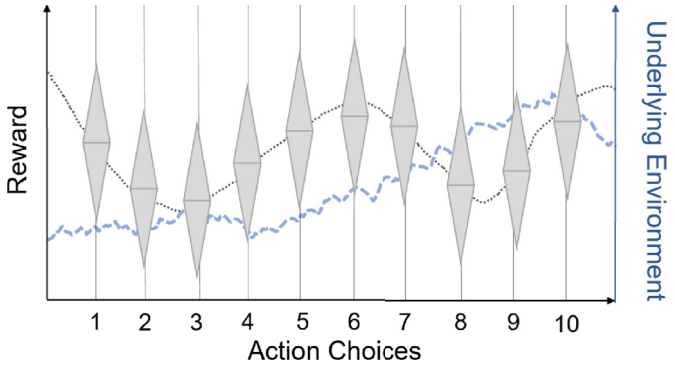


Fig. 4. A 10-armed bandit model illustration (adapted from Ref. [20]).

BW_{pred}. However, supervised training is required for this purpose. The time cost in training, optimising BW_{pred} using the trained ANN and the CPU memory usage at the CO are presented in our study [25]. Unsurprisingly, supervised training, including training the ANN and training set storage, is the main contributing factor to the time cost and CPU usage at the CO. In order to facilitate the CO to allocate bandwidth intelligently, without relying on specific BW_{pred} estimation algorithms or supervised training, we explore the potential of reinforcement learning. The problem formulation is described as follows.

2.3. From supervised learning to reinforcement learning: proposed RL-based DBA scheme

RL typically participates an *agent* and an *environment*. Different *actions* taken by the agent interact with the environment, thereby yielding different *rewards*. In RL, the agent is not told which actions to take such as that in supervised learning, but needs to estimate which actions produce the most long-term reward by trying them, and then optimises its actions. According to the principle of policy improvement in RL, any scheme that optimises its action/policy via exploration over time is anticipated to outperform schemes that take fixed actions/policies [20,21].

In this paper, we formulate the bandwidth allocation in PONs described above as a k-armed bandit problem. The k-armed bandit model is a simple and well-known RL model. Since our goal is to optimise BW_{pred} in reducing the uplink latency for low-latency applications, a simple model such as the k-armed bandit model and learning algorithms with quick convergence, which will be discussed in the following, are preferred.

In a k-armed bandit problem, an agent pulls one of the k arms in one play such as illustrated in Fig. 4, and receives an immediate reward associated with this action. This reward is a random variable and its underlying distribution is not known by the agent. Based on the rewards in previous plays, the agent determines which arm to pull for the next play, expecting that the long-term reward sum is maximised. In our problem formulation, we consider the CO as the agent and BW_{pred} is the decision to be optimised. Note that we use the term '*decision*' in this paper, which is equivalent to the 'action' in RL terminology. Upon receiving a report, the CO allocates $\min\{BW_{req} + BW_{pred}, BW_{max}\}$ amount of bandwidth to the ONU. The objective of the CO is to determine a BW_{pred} such that the average uplink latency can be reduced. As such, the reward is defined as the negative latency associated with a BW_{pred} decision. The key definitions and notations used in our model are specified as follows:

- Decision, $BW_{pred}(t)$ — the BW_{pred} determined by the CO at the t-th time step. At any $t = 0, 1, 2\ldots$, $BW_{pred}(t)$ is selected from a decision set, denoted as $\boldsymbol{D}$. This $\boldsymbol{D}$ set contains l numbers of discretised bandwidth values, i.e., $\boldsymbol{D} = \{BW_1, BW_2, \ldots BW_l\}$.
- Reward, $R(t)$ — the reward received by the CO when $BW_{pred}(t)$ is exploited, i.e., allocated to ONUs. This reward is defined as the negative packet queuing latency reported by ONUs.
- Time step, t_{step} — decision update interval. Every t_{step} polling cycles, ONUs report the $R(t)$ in their report messages. The CO selects $BW_{pred}(t+1)$ from $\boldsymbol{D}$ based on the received rewards.

Note that the CO does not have prior knowledge on the network environment and the rewards associated with each decision in $\boldsymbol{D}$. Thus, the CO needs to estimate the rewards by exploring the decisions in $\boldsymbol{D}$. As such, we consider periodic reward reporting and decision update. As described above, every t_{step} polling cycles, ONUs report the average queuing latency of packets in their buffer, i.e. $R(t)$, to the CO. Based on the received rewards, the CO selects a $BW_{pred}(t+1)$ from $\boldsymbol{D}$ for the next t_{step} polling cycles. The pseudocode of the proposed RL-based DBA scheme is detailed in Algorithm 1. The criterion to update $BW_{pred}(t+1)$ as shown in Algorithm 1 (line 12 and 13) is that the CO expects to select a BW_{pred} from $\boldsymbol{D}$ that yields maximum future rewards. In the next section, we present the algorithms for $BW_{pred}(t+1)$ decision optimisation upon the CO receiving the rewards $R(t)$.

Algorithm 1 RL-based DBA scheme.

1: **Operation at an ONU in its transmission time slot:**
2: send the buffered packets to the CO.
3: sum up packet queuing latency to d_{queue} and the number of packets sent to n_{send}.
4: **if** t = (polling cycle number/t_{step}) is integer **then**
5: average latency $d_{avg} = d_{queue}/n_{send}$.
6: $R(t) \leftarrow (-d_{avg})$.
7: reset d_{queue} and n_{send}.
8: **end if**
9: $BW_{req} \leftarrow$ remaining packets in the buffer.
10: send a report with BW_{req} and $R(t)$ to the CO.
11: **Operation at the CO upon receiving a report from an ONU:**
12: **if** $R(t)$ is updated by the ONU
13: update BW_{pred}.
14: **else**
15: retain current BW_{pred}.
16: end if
17: allocates $\min\{BW_{req} + BW_{pred}, BW_{max}\}$.

3. Bandwidth decision selection algorithms in the proposed scheme

In RL, the concept of action-value function is instrumental for the agent to select actions [20]. The action-value function indicates which action or actions can lead to the most long-term rewards, thereby guiding the agent to select an action at a particular time. This function is typically estimated by the agent via exploiting and exploring actions.[1]

Let us denote Q_t as the value function of the decisions in $\boldsymbol{D} = \{BW_1, BW_2, \ldots, BW_l\}$ at time t. For an arbitrary BW_i in $\boldsymbol{D}$, $Q_t(BW_i)$ implicates the expected future reward sum if exploiting this BW_i at time t.

In our proposed scheme, the CO estimates Q_t based on each $R(t)$ received and then selects $BW_{pred}(t+1)$ following the Bellman optimality equation as follows:

$$BW_{pred}(t+1) = \underset{BW_i \in \boldsymbol{D}}{\arg\max}\, Q_t(BW_i) \tag{1}$$

This allows the CO to iteratively optimise BW_{pred} online. The greedy decision selection in (1), i.e., exploiting the current best decision, is to

[1] Exploitation refers that an agent strategically chooses actions and obtains rewards based on existing experience. Exploration refers that an agent intentionally experiences different actions for better future action selections.

facilitate quick convergence of Q_t and $\mathrm{BW}_{pred}(t+1)$ to a steady-state solution Q and BW^*_{pred} that meet $\mathrm{BW}^*_{pred} = \arg\max Q$ [26]. Given (1), the $Q_t(\mathrm{BW}_i)(i = 1, 2, \ldots, l)$ estimation is critical in selecting decisions and impacts the performance of the proposed scheme. Several well-known methods to estimate an action-value function in RL include dynamic programming (DP), Monte Carlo (MC), temporal difference (TD), Q-learning, action-critic, etc. These methods, however, are not all suitable in solving our formulated problem. For example, DP relies on the knowledge of the underlying environment. MC requires the presence of termination states in a problem [27]. In the following, we discuss the existing methods and present the algorithms developed based on them to estimate Q_t for our proposed scheme.

3.1. Sample average algorithm

Note that up to the time step $(t+1)$, the CO observes a sequence of decisions and rewards as $\mathrm{BW}_{pred}(0)$, $R(0)$, $\mathrm{BW}_{pred}(1)$, $R(1)$, $\mathrm{BW}_{pred}(2)$, $R(2)$, …, $\mathrm{BW}_{pred}(t)$, $R(t)$. A heuristic algorithm to estimate the $Q_t(\mathrm{BW}_i)$ is by averaging the rewards tied to this BW_i in the sequence as follows:

$$Q_t(\mathrm{BW}_i) = \frac{\sum_{k=1}^{t} R(k) \times \mathbb{1}_{\mathrm{BW}_i}}{\sum_{k=1}^{t} \mathbb{1}_{\mathrm{BW}_i}} \tag{2}$$

where $\mathbb{1}_{\mathrm{BW}_i}$ is an indicator function, indicating if the BW_i is selected at one particular time step. By simplifying (2), an iterative algorithm to update Q_t for each BW_i is derived as follows:

$$Q_t(\mathrm{BW}_i) = Q_{t-1}(\mathrm{BW}_i) + \frac{R(t) - Q_{t-1}(\mathrm{BW}_i)}{\sum_{k=1}^{t} \mathbb{1}_{\mathrm{BW}_i}} \tag{3}$$

Then, based on this Q_t, $\mathrm{BW}_{pred}(t+1)$ can be selected by the CO following (1). We refer to this algorithm as sample average (SA) algorithm in this paper and its pseudocode is detailed in Algorithm 2.

Algorithm 2 SA algorithm.

1: Q — array of existing Q estimates.
2: N — array of the number of decision exploitations
3: **Operation at the CO upon receiving rewards *R(t)* at time step (*t*+1):**
4: $Q(\mathrm{BW}_{pred}(t)) \leftarrow Q(\mathrm{BW}_{pred}(t)) + \frac{R(t) - Q(\mathrm{BW}_{pred}(t))}{N(\mathrm{BW}_{pred}(t))}$.
5: $\mathrm{BW}_{pred}(t+1) = \arg\max_{\mathrm{BW}_i \in \boldsymbol{D}} Q$
6: $N(\mathrm{BW}_{pred}(t+1)) \leftarrow N(\mathrm{BW}_{pred}(t+1)) + 1$

Using the SA, the CO estimates $Q_t(\mathrm{BW}_i)$ purely relying on allocating this BW_i together with the BW_{req} to ONUs and waiting the latency to be reported. As such, exploring different BW_i is crucial for the CO to optimise the BW_{pred}. This reliance on exploration and waiting for reports tends to incur a slow convergence rate in SA. Since our focus is to improve the latency performance of the network, algorithms that enable the CO to explore decisions more efficiently, i.e., faster convergence to an optimal decision, than the SA are preferred. Therefore, we further consider TD- and Q-learning based algorithms. The design details are presented as follows.

3.2. TD-based value iteration algorithm

TD learning is an important method in RL whereby an agent learns directly from experience such as that in the SA, and concurrently updates estimates using the learnt knowledge, i.e., existing estimates. In particular, defined by (2) in the SA, $Q_t(\mathrm{BW}_i)$ can not be estimated by the CO if this BW_i has not been exploited. The received rewards of a BW_i can only be used to estimate its own $Q_t(\mathrm{BW}_i)$ value. In comparison, based on TD, the CO can iteratively updates $Q_t(\mathrm{BW}_i)$ using both received rewards $R(t)$ and existing estimates Q_t. A straightforward algorithm based on this idea is described as follow.

By definition, the value function Q_t represents the expected future rewards summation when exploiting a BW_i at time step t. As such $Q_t(\mathrm{BW}_i)$ can be re-written into (4).

$$Q_t(\mathrm{BW}_i) = E\left[\sum_{k=t}^{\infty} \gamma^{k-t} R(k) \,\middle|\, \mathrm{BW}_{pred}(t) = \mathrm{BW}_i\right] \tag{4}$$

where E denotes the expectation operator and γ is know as the discount rate $(0 \le \gamma \le 1)$. As γ approaching 1, the estimation in (4) weights strongly on future rewards, whilst a small γ accounts immediate rewards. Following (1), the next decision is $\mathrm{BW}_{pred}(t+1)$, which depends on existing Q_t. Formula (4) hence can be re-written as:

$$\begin{aligned} Q_t(\mathrm{BW}_i) &= R(t) + \gamma E\left[\sum_{k=t+1}^{\infty} \gamma^{k-(t+1)} R(k) \,\middle|\, \mathrm{BW}_{pred}(t+1)\right] \\ &= R(t) + \gamma E\left[Q_{t+1}(\mathrm{BW}_{pred}(t+1))\right] \end{aligned} \tag{5}$$

In (5), the Q_{t+1} is unknown. However, the CO can still update its estimates by trusting and using existing Q_t. Having $\mathrm{BW}_{pred}(t) = \mathrm{BW}_i$ and its corresponding $R(t)$, the CO estimates $Q_t(\mathrm{BW}_i)$ as:

$$Q_t(\mathrm{BW}_i) = R(t) + \gamma Q_t(\mathrm{BW}_{pred}(t+1)) \tag{6}$$

Since $\mathrm{BW}_{pred}(t+1)$ is selected following (1) greedily, (6) can be further derived as:

$$Q_t(\mathrm{BW}_i) = R(t) + \gamma \max Q_t \tag{7}$$

The pseudocode of the TD-based algorithm is presented in Algorithm 3. Note that the knowledge of Q_t is used to update the estimate for each BW_i in $\boldsymbol{D}$, thereby expediting the convergence of Q_t and $\mathrm{BW}_{pred}(t+1)$.

Algorithm 3 TD-based value iteration algorithm.

1: Q^* — the maximum value of existing Q estimates.
2: **Operation at the CO upon receiving rewards *R(t)* at time step (*t*+1):**
3: $Q(\mathrm{BW}_{pred}(t)) \leftarrow R(t) + \gamma Q^*$
4: $\mathrm{BW}_{pred}(t+1) = \arg\max_{\mathrm{BW}_i \in \boldsymbol{D}} Q$
5: update $Q^* \leftarrow \max Q$

3.3. Q-learning-based algorithm

Q-learning is a special TD method that is widely utilised. It is known by its quick convergence to find an optimal solution [20]. Q-learning aims to estimate the action-value function of a Markov decision process (MDP) with a series of state and action pairs, denoted as S_i and A_i respectively, and the rewards, $R(S_i, A_i)$, associated with states and actions. A multi-armed bandit model can be viewed as a MDP with a single state, S, and multiple actions. As such, we further explore the use of Q-learning in estimating Q_t for our proposed scheme.

For a single state MDP, Q-learning estimates its action-value function, termed $Q(S, A_i)$, as follows:

$$Q(S, A_i) \leftarrow (1-\alpha) Q(S, A_i) + \alpha\left[R(S, A_i) + \gamma \max_A Q(S, A)\right] \tag{8}$$

where A is the current best action and α is the learning rate, balancing the weighting between existing Q estimates and experienced rewards. Smaller α values lower the convergence rate in trading for a potential good accuracy. The arrow sign in (8) indicates overwriting $Q(S_i, A_i)$ with a new value, which is calculated by using its current value. The iteration in (8) ensures the convergence to an optimal solution when the number of states and actions are finite and the rewards have finite variance [24]. These conditions are met in our formulated model where bandwidth decisions are discretised and packet latency as reward is bounded. Following (8), $Q_t(\mathrm{BW}_i)$ can be estimated by:

$$Q_t(\mathrm{BW}_i) \leftarrow (1-\alpha) Q_t(\mathrm{BW}_i) + \alpha[R(t) + \gamma \max Q_t] \tag{9}$$

The pseudocode of Q-learning-based algorithm for Q_t estimation is presented in Algorithm 4.

Algorithm 4 Q-learning-based algorithm.

1: Q — array of existing Q estimates.
2: **Operation at the CO upon receiving rewards $R(t)$ at time step $(t+1)$:**
3: $Q(\mathrm{BW}_{pred}(t)) \leftarrow (1-\alpha)Q(\mathrm{BW}_{pred}(t)) + \alpha[R(t) + \gamma \max Q]$
4: $\mathrm{BW}_{pred}(t+1) = \arg\max_{\mathrm{BW}_i \in \boldsymbol{D}} Q$

It can be noted that the TD-based value iteration algorithm is a special case of Q-learning where α equals 1. This Q-learning based algorithm allows the CO to control the learning rate by tuning α. Note that the parameters α and γ generally are selected based on experiments or experience. Usually γ is a small value close to 0 and α is within the range of 0–1. In the next section, we implement simulations to analyse and compare the performance of the above discussed algorithms and validate the effectiveness of the proposed RL-based DBA scheme.

4. Performance evaluation

Performance analysis in this section is based on packet-driven network simulations in MATLAB. We apply the proposed scheme and the algorithms discussed above to a 10 Gbps PON with 16 ONUs and 10 km ONU-to-CO distance. Each ONU has a 1 Mbps buffer. Packet arrivals follow a Poisson process and packet length varies from 64 to 1518 bytes [18]. The maximum polling cycle duration is set as 1 ms in DBA operation. Performances of our proposed RL-based DBA scheme using above algorithms are compared with the baseline limited-service DBA scheme and statistical predictive DBA scheme (given prior knowledge of the arrival rate) presented in Section 2.1.

In implementing the RL-based scheme, we set the decision set $\boldsymbol{D}$ to include 100 number of BW_i with a 200-byte interval for illustrative purpose. This granularity is small enough to yield favourable latency improvements. The initial Q values of the decisions are set as 0 and the time step of reporting rewards is set as 5 polling cycles. The parameter γ and α are selected as 0.01 and 0.3 based on our simulation trails, and the impact of these parameters will also be analysed in the following.

4.1. Latency performance

Fig. 5 compares the latency performance under different network load scenarios, i.e., light load ($l_{network} = 0.2$), medium load ($l_{network} = 0.5$), and high load ($l_{network} = 0.8$). As shown in Fig. 5(a)–(c), the network experiences about 200–500 polling cycles to stabilise, which is less than 0.5 s in the simulation. Since using RL the CO optimises BW_{pred} iteratively in an online fashion, there is no extra time cost and memory usage for supervised training. Only the decision value $Q(\mathrm{BW}_i)$, and $N(\mathrm{BW}_i)$, the number of BW_i exploitations (when using SA algorithm), need to be stored and updated iteratively at the CO following the Algorithms in Section 3. The impact of the computation time in updating the $Q(\mathrm{BW}_i)$ per iteration on the uplink latency can be negligible. Results show that the baseline scheme yields the highest latency due to the report-and-grant process, whereby packets need to be reported before they can be allocated bandwidth for transmission. The predictive scheme estimates BW_{pred} based on packet arrival rate. This reduces the latency compared to the baseline scheme as shown in Fig. 5, but on the basis that the packet arrival rate is given in our simulation.

The proposed RL-based scheme reduces up to 50% latency compared to the baseline scheme in Fig. 5(a)–(c) when the algorithms converge. At the initial 500 polling cycles, the RL-based scheme incurs greater deviations, depending on the underlying learning algorithms. This is

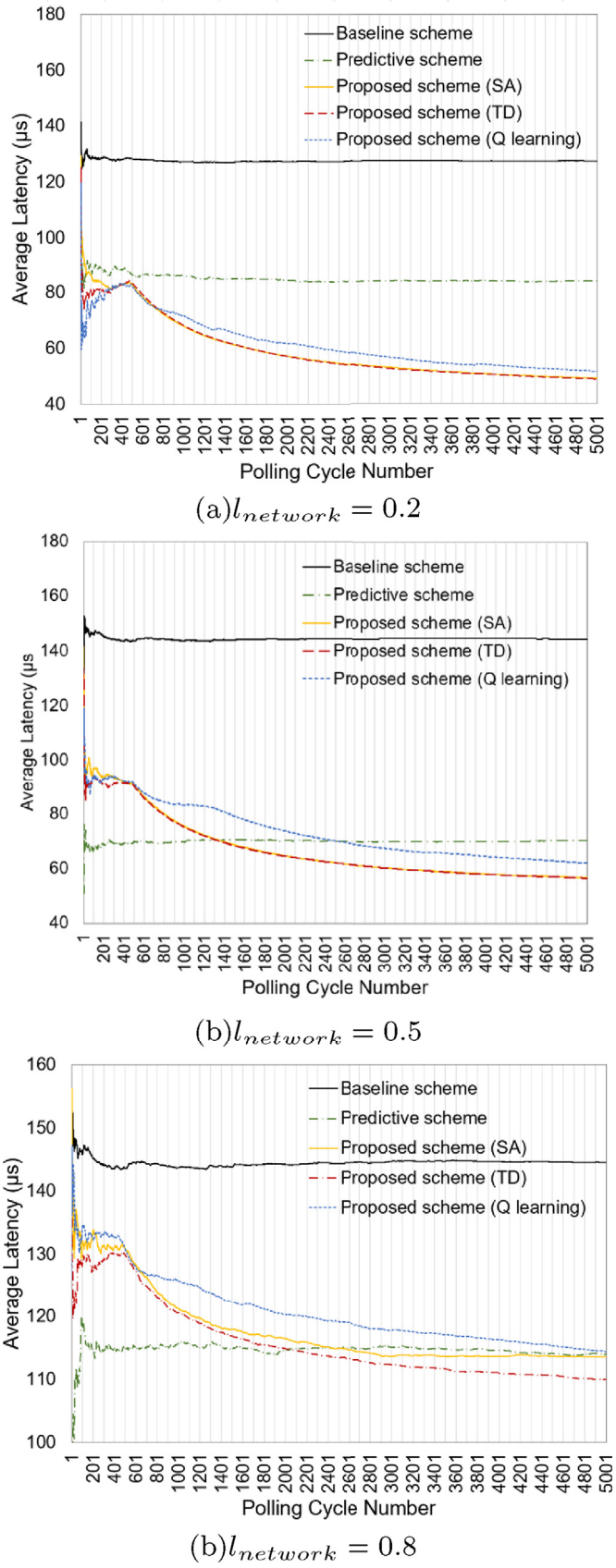


Fig. 5. Latency performance comparisons under different network loads.

because at the beginning, the CO tends to explore different decisions in order to estimate the rewards. Nevertheless, compared to the baseline scheme, the latency is still reduced effectively during this exploitative period. Then, with iterative Q_t updates, the CO adapts its BW_{pred} decisions and eventually outperforms the predictive scheme. Particularly, under light load such as $l_{network} = 0.2$ in Fig. 5(a), the proposed scheme outperforms both the baseline and predictive schemes. In comparison, the network load increase slows the average latency convergence as shown in Fig. 5(b) and (c). The RL-based scheme attains lower latency than the predictive scheme after 1400 polling cycles and 2000 polling cycles in Fig. 5(b) and (c) respectively.

In the RL-based scheme, performances of the SA, TD and *Q*-learning based algorithms are compared. As shown in Fig. 5(a) and (b), SA and TD-based algorithms have similar convergence performance, except that at the initial stage, the TD-based algorithm is slightly better. In comparison, under high load scenario in Fig. 5(c), the TD-based algorithm shows a faster convergence than the SA. This is because the TD-based algorithm updates Q_t values by using both experienced rewards and existing Q_t estimates following the Bellman optimality equation. In the SA algorithm, however, rewards are only used to estimate the value of its corresponding decisions. The *Q*-learning based algorithm can also converge to the same optimal BW_{pred}, and average network latency, as the SA and TD-based algorithm. However, the convergence rate is controlled by the α as introduced in Section 3.3, and this parameter generally needs to be determined via simulation or experimental trials. Note that the TD-based algorithm is a special case of *Q*-learning where $\alpha = 1$. In Fig. 5, we plot the *Q*-learning based algorithm using $\alpha = 0.3$ in analysing the impact of α on its performance. Clearly, a small α lowers the convergence rate as shown in Fig. 5. Since in our considered scenarios, stable network loads are simulated, applying $\alpha = 1$, i.e., the TD-based algorithm, provisions faster convergence to an optimal BW_{pred} decision. However, given the flexibility to adjust α, we consider *Q*-learning based algorithm may have the potential in more complex network scenarios such as networks with varying network loads. Overall, based on the above analysis, TD-based algorithm shows a faster convergence rate in finding an optimal decisions among the discussed algorithms. In the following, we apply the proposed RL-based scheme (TD-based algorithm) in different network scenarios and compare its latency performance with existing schemes.

4.2. RL-based scheme in different network scenarios

Fig. 6 compares the latency of the proposed scheme, baseline and predictive scheme in a 16 PON with 20 km ONU-to-CO distance. Two network load scenarios, i.e., $l_{network} = 0.2$ and $l_{network} = 0.5$ are simulated. Compared to Fig. 5, the ONU-to-CO distance increase, i.e., from 10 km to 20 km, increases the latency of the baseline scheme and predictive scheme significantly. The proposed scheme effectively prevents this network latency increase and yields superior latency performance, i.e., less than 100 μs latency, compared to the 150 μs and 250 μs latency in the predictive and baseline scheme, respectively. In addition, under $l_{network} = 0.5$, the RL-based scheme outperforms the predictive after about 700 polling cycles, which is half the time compared to that shown in Fig. 5. These observations validate that the RL-based scheme can effectively optimise the BW_{pred} to suit the underlying network environment.

Fig. 7 shows that the latency of a 32-ONU PON under different schemes. Similar latency trends can be observed compared to that in Figs. 5 and 6. The RL-based scheme converges to a BW_{pred} that reduces the latency compared to the existing schemes. However, this time, we observe a longer time for the RL-scheme to lower the latency from the predictive scheme under $l_{network} = 0.5$. This is partly because in this scenario, the BW_{pred} estimated in predictive scheme is close to the BW_{pred} selected in the RL-based scheme. The predictive scheme provisions such BW_{pred} at the beginning of the simulation, a low latency is achieved, and

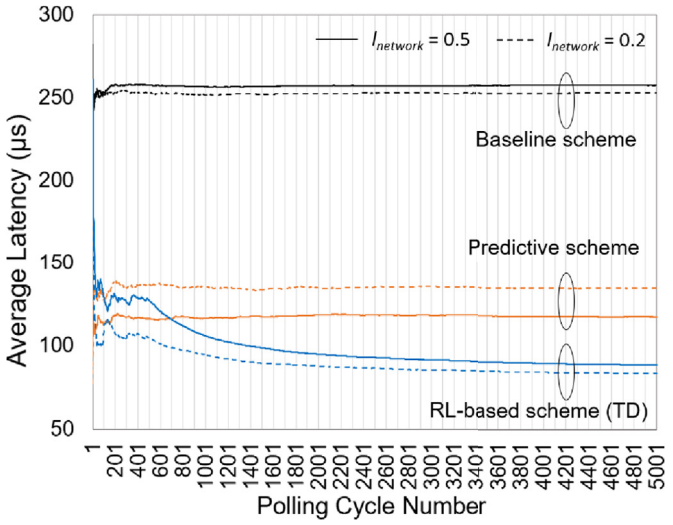


Fig. 6. Latency performance comparisons in a PON with 20 km ONU-to-CO distance.

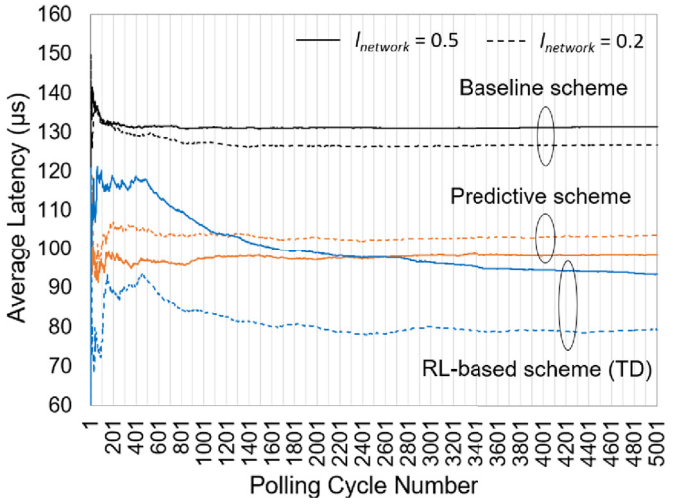


Fig. 7. Latency performance comparisons in a 10 km 32-ONU PON.

hence it takes longer time for the RL-scheme to reduce latency from this value. However, in many cases such as shown in Figs. 5 and 6, the predictive scheme can not estimate a suitable BW_{pred}, and therefore incurs a higher latency than the RL-based scheme. Moreover, the knowledge of the arrival rate is required in estimating BW_{pred} in the predictive scheme.

Overall, simulation results in Figs. 5–7 validate the effectiveness of the proposed RL-based scheme in optimising BW_{pred} and reducing latency for emerging low-latency H2M and tactile-haptic applications. Using RL, the CO self-adapts BW_{pred} decisions to best suit the underlying network environment. The allocation of optimal BW_{pred} in the RL-based DBA scheme achieves less than 100 μs uplink latency in the 16- and 32-ONU optical access network shown in Figs. 5–7, attaining up to 60% improvement compared to the baseline scheme.

5. Conclusions

Current communication networks need advanced resource allocation strategies to improve their latency in realising the emerging remotely-controlled H2M and haptic communications that demand 1–10 ms transmissions. In supporting converged applications delivery over PONs, this paper presented the first exploitation of RL to

facilitate intelligent bandwidth decisions in DBA operation at the CO. We formulated bandwidth allocation in PON as a multi-armed bandit model and proposed the RL-based scheme, whereby the CO explores and exploits bandwidth decisions and ONUs report the corresponding rewards. Then, we discussed several algorithms based on SA, TD and *Q*-learning methods in RL to iteratively estimate the decision value function. Based on the value estimation, bandwidth decisions are selected greedily in expediting algorithm convergence and reducing the latency of the network. With extensive simulations, we analysed the latency of the proposed scheme and the convergence performance of the SA, TD and *Q*-learning based algorithms. Overall, different from existing schemes that rely on specific bandwidth estimation algorithms and prior knowledge of the network, the RL-based scheme self-adaptively optimises bandwidth decisions in DBA operation, and thereby effectively reduces the latency under diverse network scenarios. A promising latency performance improvement, i.e., up to 60% reduction compared to the commonly-adopted baseline scheme, is attained towards supporting future low-latency applications.

CRediT authorship contribution statement

Lihua Ruan: Conceptualization, Methodology, Writing - original draft. **Maluge Pubuduni Imali Dias:** Writing - review & editing. **Elaine Wong:** Supervision, Writing - review & editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

[1] M. Simsek, A. Aijaz, M. Dohler, J. Sachs, G. Fettweis, 5g-enabled tactile internet, IEEE J. Sel. Area. Commun. 34 (3) (2016) 460–473.

[2] M. Maier, M. Chowdhury, B.P. Rimal, D.P. Van, The tactile internet: vision, recent progress, and open challenges, IEEE Commun. Mag. 54 (5) (2016) 138–145.

[3] M. Dohler, T. Mahmoodi, M.A. Lema, M. Condoluci, F. Sardis, K. Antonakoglou, H. Aghvami, Internet of skills, where robotics meets ai, 5g and the tactile internet, in: 2017 European Conference on Networks and Communications (EuCNC), IEEE, 2017, pp. 1–5.

[4] E. Wong, M.P.I. Dias, L. Ruan, Predictive resource allocation for tactile internet capable passive optical lans, J. Lightwave Technol. 35 (13) (2017) 2629–2641.

[5] M. De Andrade, M. Maier, M.P. McGarry, M. Reisslein, Passive Optical Network (Pon) Supported Networking, 2014.

[6] M. Maier, Fiwi access networks: future research challenges and moonshot perspectives, in: 2014 IEEE International Conference on Communications Workshops (ICC), IEEE, 2014, pp. 371–375.

[7] S. Mondal, G. Das, E. Wong, Cost-optimal cloudlet placement frameworks over fiber-wireless access networks for low-latency applications, J. Netw. Comput. Appl. 138 (2019) 27–38.

[8] A. Hmaity, M. Savi, L. Askari, F. Musumeci, M. Tornatore, A. Pattavina, Latency-and capacity-aware placement of chained virtual network functions in fmc metro networks, Opt. Switch. Netw. (2019) 100536.

[9] M. Chowdhury, M. Maier, Collaborative computing for advanced tactile internet human-to-robot (h2r) communications in integrated fiwi multirobot infrastructures, IEEE Internet Things J. 4 (6) (2017) 2142–2158.

[10] G. Kramer, B. Mukherjee, A. Maislos, Ethernet Passive Optical Networks, 2003.

[11] Z. Wang, B. Gao, N. Shiwei, L. Jing, X. Wu, Ethernet passive optical network communication method, optical network unit, and optical line terminal, uS Patent App. 16/298,677 (Jul. 4 2019).

[12] E. Wong, L. Ruan, Achieving low-latency h2m communications through predicting bandwidth demand: a comparative study of statistical prediction and machine learning techniques, in: Photonic Networks and Devices, Optical Society of America, 2019, NeTh1D4.

[13] H. Huang, T. Ye, T.T. Lee, Optimum transmission window for epons with limited service, IEEE Access 7 (2019) 57956–57971.

[14] M.P. McGarry, M. Reisslein, M. Maier, Ethernet passive optical network architectures and dynamic bandwidth allocation algorithms, IEEE Commun. Surv. Tutor. 10 (3) (2008) 46–60.

[15] M.P.I. Dias, B.S. Karunaratne, E. Wong, Bayesian estimation and prediction-based dynamic bandwidth allocation algorithm for sleep/doze-mode passive optical networks, J. Lightwave Technol. 32 (14) (2014) 2560–2568.

[16] Y. Luo, N. Ansari, Limited sharing with traffic prediction for dynamic bandwidth allocation and qos provisioning over ethernet passive optical networks, J. Opt. Netw. 4 (9) (2005) 561–572.

[17] K. Nishimoto, M. Tadokoro, T. Fujiwara, T. Yamada, T. Tanaka, A. Takeda, T. Inoue, Predictive dynamic bandwidth allocation based on the correlation of the bi-directional traffic for cloud-based virtual pon-olt, in: 2017 IEEE International Workshop Technical Committee on Communications Quality and Reliability (CQR), IEEE, 2017, pp. 1–6.

[18] L. Ruan, I. Dias, E. Wong, Machine intelligence in supervising bandwidth allocation for low-latency communications, in: 2019 IEEE 20th International Conference on High Performance Switching and Routing (HPSR), IEEE, 2019, pp. 1–6.

[19] L. Ruan, M.P.I. Dias, E. Wong, Deep neural network supervised bandwidth allocation decisions for low-latency heterogeneous e-health networks, J. Lightwave Technol. 37 (16) (2019) 4147–4154.

[20] R.S. Sutton, A.G. Barto, Reinforcement Learning: an Introduction, 2011.

[21] Y. Kiran, T. Venkatesh, C.S.R. Murthy, A reinforcement learning framework for path selection and wavelength selection in optical burst switched networks, IEEE J. Sel. Area. Commun. 25 (9) (2007) 18–26.

[22] J. Mata, I. De Miguel, R.J. Duran, N. Merayo, S.K. Singh, A. Jukan, M. Chamania, Artificial intelligence (ai) methods in optical networks: a comprehensive survey, Opt. Switch. Netw. 28 (2018) 43–57.

[23] Z.M. Fadlullah, H. Nishiyama, N. Kato, H. Ujikawa, K.-I. Suzuki, N. Yoshimoto, Smart fiwi networks: challenges and solutions for qos and green communications, IEEE Intell. Syst. 28 (2) (2013) 86–91.

[24] M. Kubat, Neural Networks: a Comprehensive Foundation by Simon Haykin, Macmillan, 1994. isbn 0-02-352781-7., The Knowledge Engineering Review 13 (4) (1999) 409412.

[25] L. Ruan, M.P.I. Dias, E. Wong, Enhancing latency performance through intelligent bandwidth allocation decisions: a survey and comparative study of machine learning techniques, IEEE/OSA J. Opt. Commun. Netw. 12 (4) (April 2020) B20–B32.

[26] M. Garavello, P. Soravia, Optimality principles and uniqueness for bellman equations of unbounded control problems with discontinuous running cost, Nonlinear Differ. Equ. Appl. NoDEA 11 (3) (2004) 271–298.

[27] L. Busoniu, R. Babuska, B. De Schutter, D. Ernst, Reinforcement Learning and Dynamic Programming Using Function Approximators, CRC press, 2017.