UNITED STATES PATENT APPLICATION

For

INTELLIGENT BANDWIDTH ALLOCATION IN ACCESS NETWORK BY USING MACHINE LEARNING FOR LATENCY MANAGEMENT

INVENTORS:

JINGJIE ZHU

JUNWEN ZHANG

QI ZHOU

ZHENSHENG JIA

BERNARDO HUBERMAN

Intelligent Bandwidth Allocation in PON by using Machine Learning for Latency Management

Version 1

Jingjie Zhu, Junwen Zhang, Qi Zhou, Zhensheng Jia, and Bernardo Huberman

Background and Motivation

Delivering more bandwidth/capacity has been the top research focus in optical access network, however, new services like 5G mobile X haul, edge computing, AR/VR Gaming, Tactile Internet and UHD video distribution, are placing additional requirements on access networks. Characteristics like low latency and reliability will be increasingly important for future access networks. As we move to the next-generation of access networks, ultra-low latency transmission is increasingly gaining importance in access networks for emerging time critical services. More deterministic and reliable latency management are being demanded.

Different services have different latency requirements, for instance, Cloud Gaming requires <40-ms latency for good experience, Ultra-Reliable Low Latency Communication requires <1-ms e2e latency, while Enhanced Mobile Broadband would be more relax on delay (about 100ms latency). Based on these requirements, the latency contributed by optical access network, which could be the “last-mile” carrier for these services, can be stricter. For instance, a 1-10 ms e2e latency is required for F1 mobile fronthaul interface, while this number reduces to only few 100 μs (<1ms) if we move to lower layer function-split for mobile fronthaul.

As a point-to-multi-point system, Passive Optical Network (PON) has been one of the dominant architectures to provide bandwidth sharing between different types of services [1]. In generally, dynamic bandwidth allocation (DBA) is used in PON to allocate traffic bandwidth in upstream based on the demands and requests from users (ONUs). Different DBA algorithms or strategies has been proposed to support the upstream bandwidth sharing [2], i.e., Interleaved Polling with Adaptive Cycle Time (IPACT) is widely studied and used for Ethernet PON, linear fixed bandwidth allocation with high priority is proposed for mobile PON that support mobile fronthaul services. Some DBA algorithms differentiate users into groups that support guaranteed bandwidth or non-guaranteed bandwidth. Other DBA schemes with different QoS tiers are also supported in ITU-T PONs.

Theoretically, different DBA algorithms would be suitable for different use scenarios or traffic conditions. In addition, the “optimal” DBA scheme for the same network can vary from time to time as the traffic load during a day or week can change dramatically. When considering the network delay, different users/service may have different latency requirements. When traffic loads from each user changes, the corresponding network latency also changes. As mentioned above, many emerging services require more deterministic and reliable latency. Furthermore, it is desired to achieve higher network efficiency when the network when maintaining low latency for specific users. Therefore, an intelligent bandwidth allocation that can perceive or sense the network environment changes and correspondingly updates its bandwidth allocation policy smartly to manage the latency for different users would be very attracting.

This disclosure describes a novel method for intelligent bandwidth allocation in PON by using machine learning for latency management. The specific method uses reinforcement learning scheme to proactive update the bandwidth allocation policy to control or manage the latency for specific users. The proposed method uses the traffic information as well as network parameters as the *State* input to the core control unit *Agent*. The *Agent* updates the bandwidth allocation policy accordingly, which will be the output *Actions* to the OLT in PON. These *Actions* may include updates of the network bandwidth allocation algorithms, adjusting ONU priorities or changing the bandwidth allocation parameters and so on. As a result, the PON will measure the network performance such as latency, throughput or efficiency and return these results to the *Agent* as the *Reward* feedbacks. The *Reward* can be positive or negative based on the results. The training process is based upon the input of the *Rewards*, and the reinforcement learning algorithms will decide to reward or punish the model based on its output *Actions*.

Invention Idea

The general idea of the disclosure is shown in Fig. 1, in which we describe the problem and the solution for intelligent bandwidth allocation in PON.

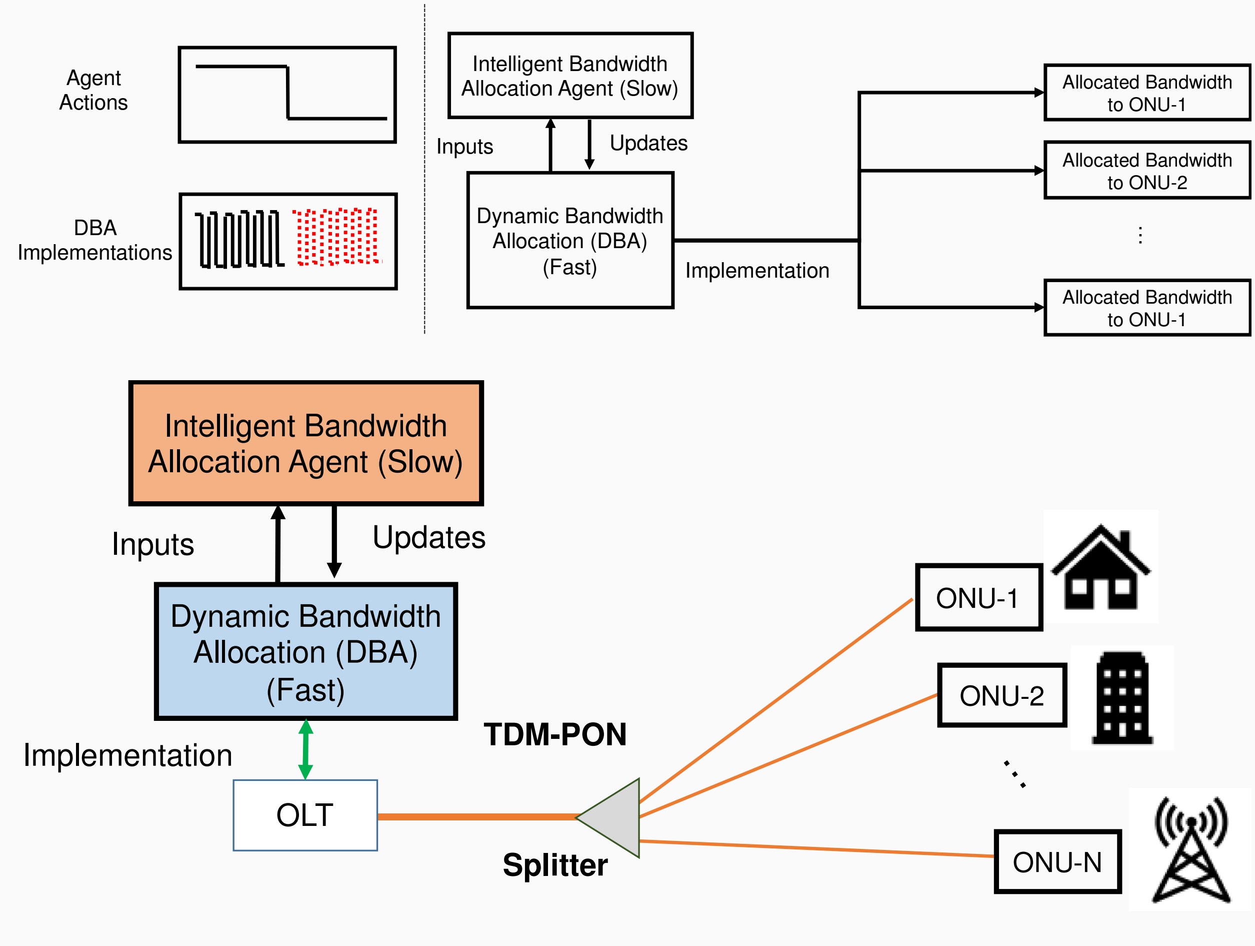

Agent Actions
DBA Implementations
Intelligent Bandwidth Allocation Agent (Slow)
Inputs
Updates
Dynamic Bandwidth Allocation (DBA) (Fast)
Implementation
Allocated Bandwidth to ONU-1
Allocated Bandwidth to ONU-2
Allocated Bandwidth to ONU-1
Intelligent Bandwidth Allocation Agent (Slow)
Inputs
Updates
Dynamic Bandwidth Allocation (DBA) (Fast)
Implementation
OLT
TDM-PON
Splitter
ONU-1
ONU-2
ONU-N

Fig. 1. A two-layer implementation by proposed intelligent bandwidth allocation in PON with different time scales. Then implementation in PON is also plotted here.

In our invention disclosure, the proposed intelligent bandwidth allocation in PON is realized by a two-layer implementation structure. In the lower layer, fast dynamic bandwidth allocation is implemented in µsec to msec scale for real-time bandwidth assignments according the actual bandwidth request from ONUs. On higher layer, we have the intelligent bandwidth allocation agent to update the DBA algorithm (policy) in the sacle of hundred msec to sec and even minutes range based on the inputs from the network. The update actions are implemented in a non-real-time manner, which is based on the reinforcement learning algorithms.

A more detailed implementation for intelligent bandwidth allocation agent is shown in Fig. 2. Reinforcement learning (RL) has demonstrated prominent performance in strategy selection and optimization tasks, such as AlphaGo [3], professional gaming [4] and interference avoidance [5], to name a few. The RL agent can obtain positive/negative reward on its executed action under a certain state through interaction with the environment.

The intelligent bandwidth allocation is realized by the reinforcement learning process, in which consists of an intelligent bandwidth allocation *Agent*, the PON *Environment* and the implementation links (*State, Action and Reward*). The proposed method uses the traffic information as well as network parameters as the *State* S_t input to the core control unit *Agent*. The *Agent* updates the bandwidth allocation policy accordingly, which will be the output *Action* A_t to the OLT in PON. These *Action* A_t may include updates of the network DBA algorithms, strategies, adjusting ONU priorities or changing the bandwidth allocation parameters and so on. As a result, the PON will implement the updated DBA policy and measure/monitor the network performance such as latency, throughput or efficiency and return these results to the *Agent* as the *Reward* R_t feedbacks. The *Reward* can be positive or negative based on the results. The training process is based upon the input of the *Rewards*, and the reinforcement learning algorithms will decide to reward or punish the model based on its output *Action* A_t. In such an implementation flows, the S_t, A_t *and* R_t may contain multiple variables or a set of elements. The training process may take multiple iterations or be implemented continuously with online learning.

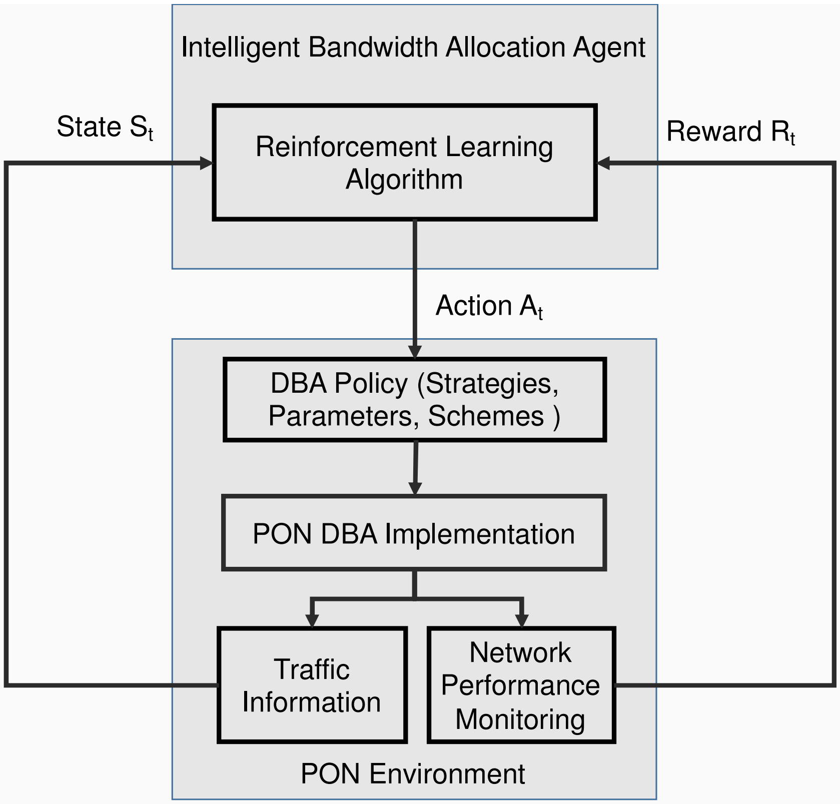
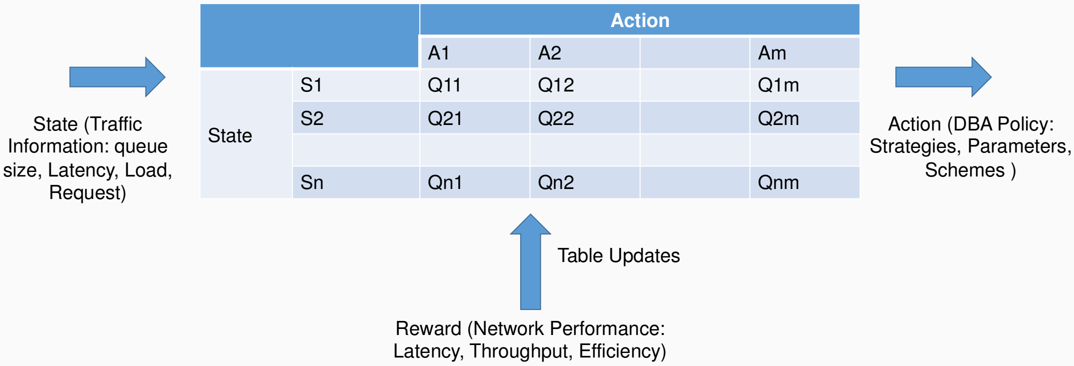

Intelligent Bandwidth Allocation Agent
State S_t
Reinforcement Learning Algorithm
Reward R_t
Action A_t
DBA Policy (Strategies, Parameters, Schemes)
PON DBA Implementation
Traffic Information
Network Performance Monitoring
PON Environment

Fig. 2 The implementation process flow of reinforcement learning for intelligent bandwidth allocation of PON

Implementation

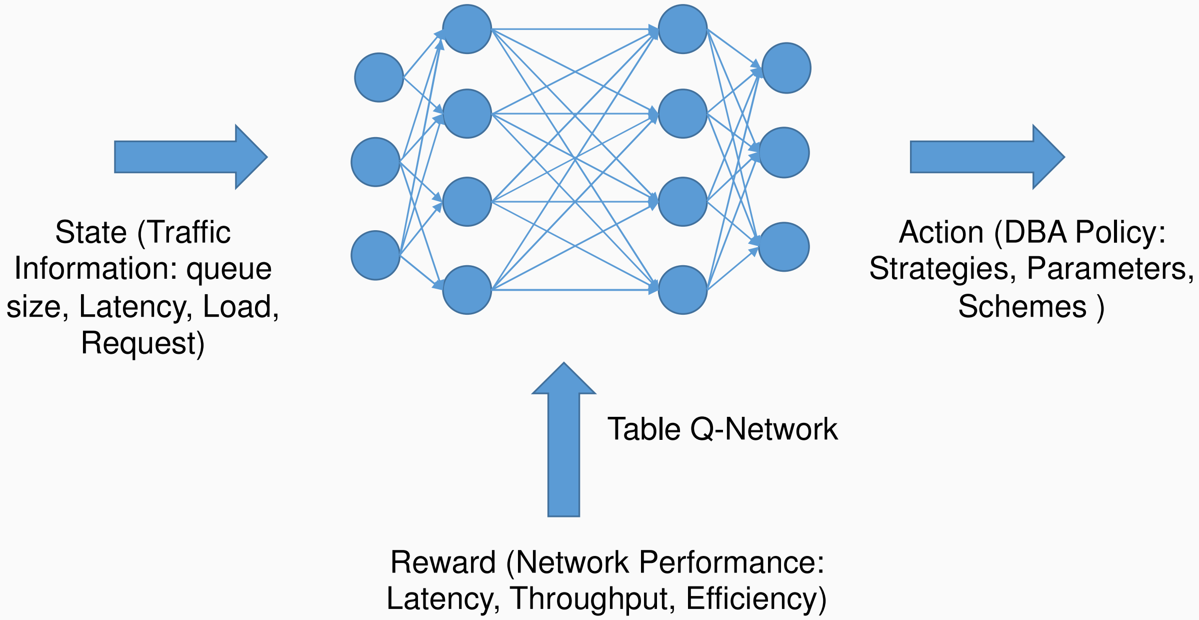
In general, there are two different implantation reinforcement learning algorithms, including table-based Q-learning and network-based Deep Q-Network. Fig. 3 and 4 show two implementation examples as Q-learning and Deep Q-Network. Q-learning is a simple yet quite powerful algorithm to create Q-value table for the operation Agent; while deep Q-learning use a multi-layer neural network to approximate the Q-value function when size of Q-value table is too large or Q-table is not available. The state is given as the input for the neural network and the Q-value of all possible actions is generated as the output.



		Action			
		A1	A2		Am
State	S1	Q11	Q12		Q1m
	S2	Q21	Q22		Q2m
	Sn	Qn1	Qn2		Qnm

State (Traffic Information: queue size, Latency, Load, Request)
Action (DBA Policy: Strategies, Parameters, Schemes)
Table Updates
Reward (Network Performance: Latency, Throughput, Efficiency)

Fig. 3. Table-based Q-learning method



State (Traffic Information: queue size, Latency, Load, Request)

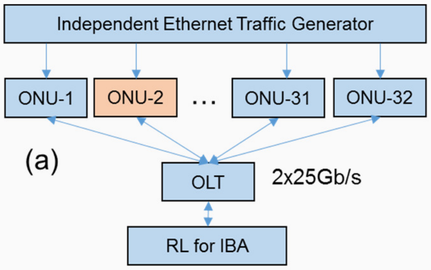
Action (DBA Policy: Strategies, Parameters, Schemes)

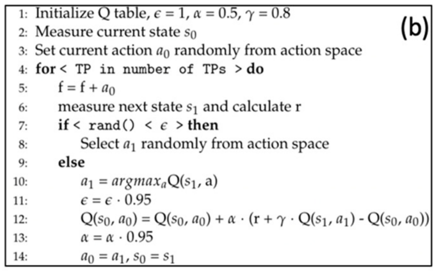
Table Q-Network

Reward (Network Performance: Latency, Throughput, Efficiency)

Fig. 4 The Q-network based deep Q-network method

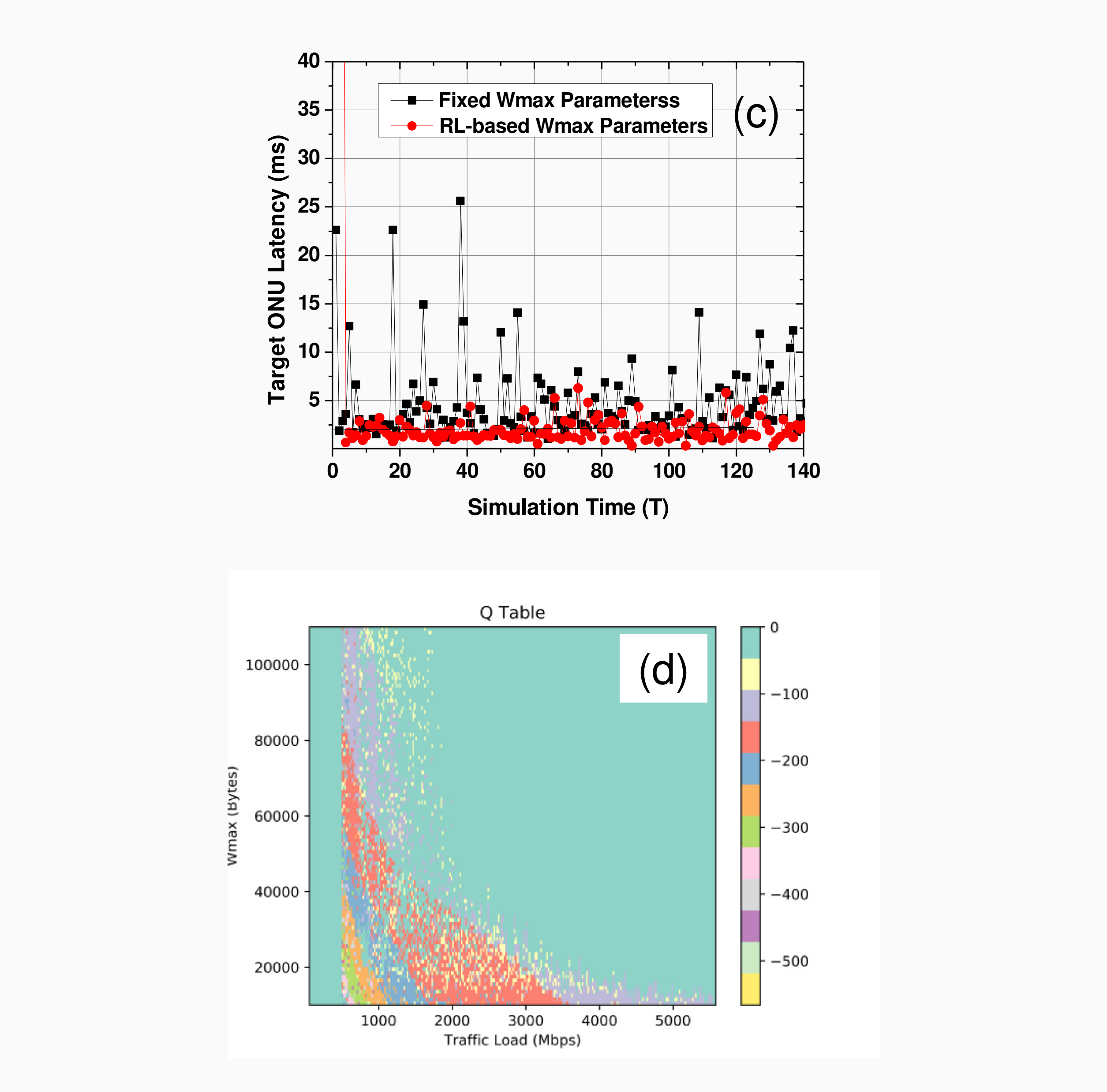
Examples





(b)

```
Initialize Q table, $\epsilon = 1$, $\alpha = 0.5$, $\gamma = 0.8$
Measure current state $s_0$
Set current action $a_0$ randomly from action space
for < TP in number of TPs > do
    $f = f + a_0$
    measure next state $s_1$ and calculate r
    if < rand() < $\epsilon$ > then
        Select $a_1$ randomly from action space
    else
        $a_1 = argmax_a Q(s_1, a)$
        $\epsilon = \epsilon \cdot 0.95$
        $Q(s_0, a_0) = Q(s_0, a_0) + \alpha \cdot (r + \gamma \cdot Q(s_1, a_1) - Q(s_0, a_0))$
        $\alpha = \alpha \cdot 0.95$
        $a_0 = a_1, s_0 = s_1$
```


(c)
Fixed Wmax Parameterss
RL-based Wmax Parameters
Target ONU Latency (ms)
40
35
30
25
20
15
10
5
0
20
40
60
80
100
120
140
Simulation Time (T)
Q Table
(d)
Wmax (Bytes)
100000
80000
60000
40000
20000
1000
2000
3000
4000
5000
Traffic Load (Mbps)
0
−100
−200
−300
−400
−500

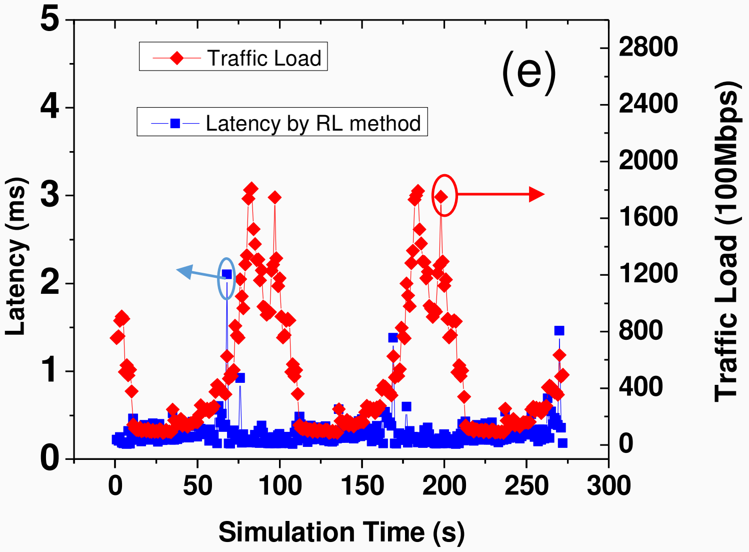


Fig. 5. The simulation setup and results: (a) the simulation setup; (b) the SARSA algorithm used in the simulation; (c) simulation of RL to target 3-ms latency; (d) the Q-table value obtained; (d) the latency performance with dynamic traffic load.

Figure 5 shows one implementation example with simulation setup and results obtained. We simulated 32 ONUs in the NG-EPON system with two wavelengths each carrying 25 Gb/s data, as shown in Fig. 3 (a). We assume 1 μs for guard interval time according to our experimental verification above. As such, a total of 50-Gb/s capacity is shared by the 32 ONUs using first-fit scheme for DBA on the two wavelengths. All 32 ONUs have random RTTs within the range of 100 to 200 μs. ONU2 is the target ONU that is enabled with latency management based on RL method. All traffic is generated by an Ethernet traffic generator model that is described in [6], where self-similar traffic is generated based on the aggregation of multiple streams, each consisting an alternating Pareto-distributed ON/OFF period [6]. The Ethernet traffics are with the packet size of 64 to 1518 bytes, and maximum traffic load for each ONU is 2 Gb/s. The default Wmax for simulation is set at 30000 bytes. The Q-table update interval and Wmax adjustment interval are all set as 0.8 s.

The algorithm for the implemented SARSA learning is designed as shown in Fig. 5 (b). Fig. 5 (c) shows the latency management results at the fixed load rate of 1.0 at 2 Gb/s. To verify the latency management capability, we set two target latency values at 3 ms and 1 ms. It is seen that in result of Fig. 5 (c) the target ONU2 follows the latency targets < 3 ms and < 1 ms with our latency management. As a comparison, we plot the latency performance under a fixed Wmax setting to present that the variance of latency is significantly reduced by employing latency management.

Fig. 5 (d) shows the Q value distribution of the Q-table after training with 1-ms target latency and different traffic loads, we can see that the peak data rate of upstream burst traffic can be as high as 5.5 Gb/s. Finally, the latency management performance with dynamic traffic loads is shown in Fig. 5 (e). By simulating the traffic load changes based on the trend obtained from a real user traffic behavior during a day, with a target latency of 1 ms, the determinism and reliability of latency management are demonstrated by the simulation result that the average latency of ONU2 is below 1 ms, with a peak latency about 2 ms due to bursty traffic.

Reference:

[1] F Effenberger, et al., IEEE Communications Magazine 45 (3), S17-S25 (2007).

[2] B. Skubic, et al., Convergence of Mobile and Stationary Next-Generation Networks, 227-252 (2010).

[3] J. X. Chen, Comput. Sci. Eng. 18, 4 (2016).

[4] V. Mnih, et al., Nature 518, 529 (2015).

[5] Q. Zhou, et al., Opt. Lett. 44, 4347-4350 (2019).

[6] G. Kramer, et al., IEEE Communications Magazine, 40. 74-80 (2002).