Received November 16, 2020, accepted November 26, 2020, date of publication December 1, 2020,
date of current version December 11, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3041785

Flexible Multi-Thread Dynamic Bandwidth Allocation Algorithm in VPONs Based on LR WDM/TDM PON

LIJUAN WU, CHAOQIN GAN, XINGDI WANG, ZHONGSEN XU, AND JIANQIANG HUI
Key Laboratory of Specialty Fiber Optics and Optical Access Networks, Joint International Research Laboratory of Specialty Fiber Optics and Advanced Communication, Shanghai University, Shanghai 200072, China

Corresponding author: Chaoqin Gan (cqgan@shu.edu.cn)

This work was supported in part by the Science and Technology Commission of Shanghai Municipality under Grant 20511102400, and in part by the 111 project under Grant D20031.

ABSTRACT With the smooth upgrade of the access network, multiple tuning-time devices will coexist in the long-reach access network, and the subsequent problems of high round-trip time (RTT) and optical network unit (ONU) tuning delay need to be solved urgently. In this paper, a multi-thread multiple tuning-time devices coexistence (MT-MTDC) bandwidth allocation algorithm is proposed. This algorithm can solve the problems of high RTT and ONU tuning delay in virtual passive optical network (VPON) based on long-reach wavelength division multiplexing / time division multiplexing passive optical network (LR WDM/TDM PON). Firstly, Multi-threaded polling mechanism is introduced into multi-mode coexistence VPON. Next, the number of wavelengths and threads is selected adaptively to ensure high bandwidth utilization. Then, the mechanism of dynamically adjusting the thread window is proposed. The mechanism strengthens the collaboration ability between threads and solves the degradation problem of the multi-thread algorithm. Furthermore, the construction of tuning buffer and the setting of time flag effectively solve the problem of more frequent ONU tuning pressure and ONU transmission conflict caused by multi-threaded polling algorithm. Finally, by comparing with the multi-thread longest-first first-available (MT-LFFA) algorithm and the multi-tuning-time ONU scheduling (MOS) algorithm, the proposed algorithm demonstrates its effectiveness in terms of polling cycle time, average tuning delay, bandwidth utilization and average packet delay.

INDEX TERMS Multi-thread, round-trip time, flexible thread window, tuning delay, virtual passive optical network.

I. INTRODUCTION

With the emergence of 5G, the number of users and the demand of network are gradually increasing, and the distribution scope of users is gradually expanding [1]. In order to enlarge the coverage of access network and increase the number of users that can be loaded, long-reach access network technologies have appeared one after another, such as: long reach passive optical network (LR-PON) [2]–[6], long reach wavelength division multiplexing / time division multiplexing passive optical network (LR WDM/TDM PON) [7]–[10]. These access network technologies can increase the maximum distance between the optical line terminal (OLT) and optical network unit (ONU) to 100 km, so that the access network can carry more users. Virtual passive optical network (VPON) is an open network architecture. In the access network, ONUs establish virtual connections with other OLTs to share wavelengths, which will form the so-called VPON [11], [12]. In VPON based on LR PON or LR WDM/TDM PON, the ONU in the heavy-load OLT subsystem can be added to the light-load OLT subsystem by switching the operating wavelength of the ONU. Therefore, excellent load balancing performance can be achieved.

In VPON, whether it is the uplink transmission within the subsystem based on WDM/TDM or the load balancing among multiple subsystems, the wavelength tuning of ONU needs to be considered. How to reduce the tuning time of ONU is one of the research focuses of bandwidth allocation

The associate editor coordinating the review of this manuscript and approving it for publication was Qunbi Zhuge.

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

TABLE 1. Algorithm comparison.

Representative article	Solve high RTT problem	When the ONU tuning time is not zero, no additional delay is generated	When multiple tuning time ONUs coexist, no additional delay is generated
[17, 18]	×	×	×
[2, 19]	√	×	×
[13, 20]	×	√	×
[8, 21]	√	√	×
[15]	×	√	√

algorithm for VPON. At the same time, according to the evolution mode of upgrading on demand in access network, ONU devices within PON have different performances, such as tuning range and tuning time. Therefore, there will be multi-mode ONU coexistence in VPON based on long-reach access network. However, the existing dynamic bandwidth allocation algorithm that only considers the unified ONU tuning time is not applicable to the multi-mode coexistence VPON based on the long- reach access network [8], [13].

In addition, the round-trip time (RTT) between OLT and ONU is very high in multi-mode coexistence VPON based on long-reach access network. The existing single-threaded polling algorithm for VPON is not suitable for multi-mode coexistence VPON based on long-reach access networks [14], [15]. Compared with single thread, the multi-thread scheme is more suitable for VPON based on LR-PON [16]. By building multiple transmission channels between OLT and ONU, the bandwidth allocation algorithm based on multi-threaded polling mechanism can take advantage of idle time slots caused by high RTT [3]. Combining the factors of ONU tuning time and high RTT, authors investigated and classified the existing bandwidth allocation algorithms, and selected representative literatures for comparison. The results are shown in Table 1.

From Table 1, it can be found that for the multi-mode coexistence VPON based on long-reach access network, the dynamic bandwidth allocation algorithm which can simultaneously solve the problems of high RTT, non-zero tuning time and coexistence of multiple tuning devices has not been studied. To solve this problem, this paper will propose a multi-thread multiple tuning-time devices coexistence (MT-MTDC) bandwidth allocation algorithm. The MT-MTDC algorithm is based on the offline polling mode and can be applied to VPON based on LR WDM/TDM PON. This algorithm can not only effectively solve the problem of idle time slot caused by high RTT and problems of transmission conflict and wavelength tuning delay in the scene of multiple tuning-time devices coexistence, but also ensure the load balancing and high bandwidth utilization.

This paper is structured as follows. In section II, the MT-MTDC bandwidth allocation algorithm is presented. In section III, the slot scheduling of MT-MTDC algorithm is introduced. The simulation results are displayed to confirm the rationality and effectiveness of the proposed algorithm in section IV. Finally, this paper is concluded in section V.

II. MT-MTDC BANDWIDTH ALLOCATION ALGORITHM

A. ADAPTIVE THREAD NUMBER AND WAVELENGTH NUMBER SELECTION MECHANISM

In multi-thread multiple tuning-time devices coexistence (MT-MTDC) algorithm, the number of threads in a polling cycle is not fixed. This is because the idle slots caused by high RTT cannot be fully utilized when the number of threads is too small. When there are too many threads, the frequent communication between ONU and OLT will lead to the waste of bandwidth resources and increase the tuning times of ONU. Therefore, the number of threads will be dynamically adjusted according to the current load in the MT-MTDC algorithm.

According to the requested bandwidth of ONUs, the total number of threads required to transfer data while making full use of the idle time slots caused by RTT is calculated by:

$$Thread_{num} = \left\lceil \frac{n_{wave}^{active} * RTT_{max} * w_i}{B_{request}} \right\rceil + 1 \qquad (1)$$

where n_{wave}^{active} is the number of wavelengths actually used in VPON, RTT_{max} is the maximum RTT time of all ONUs in VPON, w_i is the wavelength bit rate, $B_{request}$ is the total bandwidth requested by all ONUs in a single thread. $n_{wave}^{active} * RTT_{max} * w_i$ represents the total amount of wasted bandwidth caused by RTT on all working wavelengths in VPON. $\left\lceil \frac{n_{wave}^{active} * RTT_{max} * w_i}{B_{request}} \right\rceil$ represents the number of threads required to make full use of the idle time slots caused by RTT. Parameter 1 is the original thread used to transmit the requested bandwidth of ONUs in VPON.

In VPON scenario based on LR WDM/TDM PON, when the load is low, bandwidth resources can be saved by reducing the number of used wavelengths. According to (1), it can be concluded that the more wavelengths enabled under the same load, the more threads required to fully use idle time slots caused by RTT. The increase in the number of threads will intensify the wavelength-tuning times of ONU and increase the scheduling pressure of the algorithm. Therefore, it is necessary to determine the optimal number of threads and wavelengths. First, traverse the number of wavelengths. Then, the number of threads calculated by (1) is used to find the optimal number of wavelengths enabled. The detailed pseudo-code of the MT-MTDC algorithm for selecting the optimal enabled wavelength number is shown in Table 2. The purpose of the step 4 is to find the best combination of

TABLE 2. Find the optimal number of active wavelengths.

Algorithm 1: Find the optimal number of active wavelengths

Input: $Thread_{num}$, $wave_{num}$, T_{cycle}, w_i, $B_{request}$, RTT_{max}
Output: n_{wave}^{active}
1: function Find($Thread_{num}$, $wave_{num}$, T_{cycle}, w_i, $B_{request}$, RTT_{max})
2: for $n_{wave}^{active} = wave_{num} \rightarrow 1$ **do**
3: **let** $Thread_{num}$ be assigned value through **equation (1)**
4: **if** $Thread_{num} * B_{request} / (T_{cycle} * w_i)! = n_{wave}^{active}$ **and** $Thread_{num} * B_{request} < T_{cycle} * w_i * wave_{num}$ **do**
5: **continue**
6: **else do**
7: **break**
8: return n_{wave}^{active}
9: end function

threads number and active wavelengths number based on the relationship between the number of threads and the number of wavelengths working in the VPON. The first formula in step 4 is to determine whether the number of active wavelengths is appropriate. In the formula, $Thread_{num} * B_{request}$ represents the total bandwidth that can be transmitted by all threads in a polling cycle, and $T_{cycle} * w_i$ represents the bandwidth that can be transmitted by a single wavelength in a polling cycle. $\frac{Thread_{num} * B_{request}}{T_{cycle} * w_i}$ indicates the number of wavelengths that should be activated. When the calculated result is not equal to the number of wavelengths operating in the VPON, it indicates that the number of wavelengths is inappropriate, and the number of operating wavelengths should continue to be modified. The second formula is to ensure that the total bandwidth of all threads does not exceed the total bandwidth provided by all wavelengths in VPON. If the number of active wavelengths is not appropriate and the total bandwidth of all threads does not exceed the total bandwidth provided by all wavelengths in VPON, the algorithm will continue to traverse the number of wavelengths until the optimal number of threads and wavelengths is found.

In the process of bandwidth allocation based on the principle of fairness, the bandwidth allocated to the ONU in each thread should not be less than the minimum guaranteed bandwidth. Each thread has a minimum transmission bandwidth. So, the number of threads is constrained when the polling cycle is limited. According to the minimum guaranteed bandwidth of a single thread, the maximum number of threads in a polling cycle can be determined by:

$$Thread_{max} = \frac{n_{wave}^{active} * T_{cycle} * w_i}{N * (B_g + T_g)} \tag{2}$$

where T_{cycle} is the polling cycle time, N is the number of ONUs in the VPON, B_g and T_g respectively represent the minimum guaranteed bandwidth of each ONU and the required guard time slot bandwidth for ONU transmission.

B. DYNAMIC ADJUSTMENT OF THREAD WINDOW

In the multi-threaded polling algorithm, when the bandwidth difference between two threads is too large, the multi-threaded algorithm will be distorted and cannot effectively use the idle time slot. In severe cases, the multi-threaded algorithm will degenerate into a single-threaded algorithm. Therefore, ideally, the amount of bandwidth transmitted by each thread should be kept the same. However, considering the volatility of ONU bandwidth request and the flexibility of VPON networking, thread window should be flexible to enhance the cooperation between threads. When a thread needs more bandwidth, the transmission window size of the thread can be increased appropriately. In the subsequent threads, the bandwidth that has been transmitted in advance within the same polling cycle is compensated to ensure that the overall allocation bandwidth in each polling cycle does not exceed the standard. When a thread needs less bandwidth, it can allocate its remaining bandwidth to subsequent threads with greater demand.

First, according to the number of threads and enabled wavelengths, the standard bandwidth window size of a single thread can be calculated:

$$Thread_{standard}^{window} = \frac{n_{wave}^{active} * T_{cycle} * w_i - N * T_g * Thread_{num}}{Thread_{num}} \tag{3}$$

where $n_{wave}^{active} * T_{cycle} * w_i$ represents the total bandwidth of all operating wavelengths in the VPON in a polling cycle, $N * T_g * Thread_{num}$ represents the total protection slot bandwidth of all threads. The subtraction result represents the total bandwidth available for ONU data transmission in a polling cycle. Divided by the number of threads, it represents the standard bandwidth of each thread.

In order to ensure that the subsequent bandwidth window cannot be too small, there is a limit to the expansion of the bandwidth window of a single thread. The minimum bandwidth window of a single thread is limited to:

$$Thread_{min}^{window} = \max[N * (B_g + T_g), \frac{n_{wave}^{active} * RTT_{max} * w_i}{Thread_{num} - 1}] \tag{4}$$

where $N * (B_g + T_g)$ is the minimum value of guaranteed bandwidth and protected slot bandwidth required to transmit all ONUs in a single thread, $\frac{n_{wave}^{active} * RTT_{max} * w_i}{Thread_{num} - 1}$ derived from (1) and represents the minimum bandwidth value that each thread must transmit in order to make full use of the idle time slots caused by RTT. Here, the largest value in the above two formulas is the minimum bandwidth window of a single thread.

Due to the variable number of threads and the sudden change of ONU bandwidth request, any thread may apply for excess bandwidth. And it is also possible that one thread has extra bandwidth available for subsequent threads. Therefore, a "borrowing" idea is introduced to expand and compensate the bandwidth of threads. At the beginning of each polling cycle, the total pre-payable bandwidth of each thread is set to:

$$B_{boundary} = Thread_{standard}^{window} - Thread_{min}^{window} \tag{5}$$

At the same time, a pre-payable bandwidth B_{loan} is also set to monitor whether the allocated thread bandwidth is exceeded in the current polling cycle. B_{loan} is equivalent to a buffer pool to supplement bandwidth for each thread or store the extra bandwidth of each thread. In the initial state, the value of B_{loan} is the maximum amount of bandwidth that can be advanced by each thread, that is, $B_{boundary}$. After that, the value of B_{loan} decreases or increases with the amount of bandwidth required by each thread. When the bandwidth required by a thread exceeds $Thread^{window}_{standard}$, the thread needs to borrow bandwidth from other threads, and the borrowed bandwidth is deducted from the B_{loan}. When the required bandwidth of the thread is less than $Thread^{window}_{standard}$, the thread will have surplus bandwidth, and the surplus bandwidth will be added to B_{loan}.

In each polling cycle, the allocation strategy of the last thread is different from that of other threads. If the previous threads use pre-payable bandwidth, it will be compensated in the last thread.

For the non-last thread, the final bandwidth allocation is as follows:

$$Thread^{window} = \begin{cases} Thread^{window}_{standard} + B_{loan} & if\ Condition1 \\ B_{request} & if\ Condition2 \\ Thread^{window}_{standard} & if\ Condition3 \\ B_{request} & if\ Condition4 \end{cases} \tag{6}$$

Attached:

$$\begin{aligned} &\textbf{\textit{Condition 1}}: B_{request} > Thread^{window}_{standard}\ \textbf{\textit{and}} \\ &\qquad B_{loan} > 0\ \textbf{\textit{and}} \\ &\qquad Thread^{window}_{standard} + B_{loan} < B_{request} \end{aligned}$$

$$\begin{aligned} &\textbf{\textit{Condition 2}}: B_{request} > Thread^{window}_{standard}\ \textbf{\textit{and}} \\ &\qquad B_{loan} > 0\ \textbf{\textit{and}} \\ &\qquad Thread^{window}_{standard} + B_{loan} \geq B_{request} \\ &\textbf{\textit{Condition 3}}: B_{request} > Thread^{window}_{standard}\ \textbf{\textit{and}}\ B_{loan} = 0 \\ &\textbf{\textit{Condition 4}}: B_{request} \leq Thread^{window}_{standard} \end{aligned}$$

where B_{loan} is calculated by:

$$B_{loan} = B_{loan} - \left(Thread^{window} - Thread^{window}_{standard}\right) \tag{7}$$

Take Condition 2 as an example to describe the scenario, and the remaining decision scenarios are similar. Condition 2 refers to that when the bandwidth request of a thread is greater than the standard bandwidth, the thread requests to expand its capacity. The current pre-payable bandwidth is greater than zero. And the standard bandwidth plus the pre-paid bandwidth can meet the needs of this thread. Therefore, the requested bandwidth is allocated to the thread, and the amount of prepaid bandwidth will be deducted from B_{loan} later.

Once the bandwidth allocation of a thread is determined, the value of B_{loan} will be updated according to (7). When the thread demand is less than the standard bandwidth window size, B_{loan} will increase according to (7). This indicates that the remaining available bandwidth will be added to the pre-payable bandwidth (B_{loan}) for the bandwidth allocation of subsequent threads.

For the last thread in each polling cycle, the bandwidth allocation is as follows:

$$Thread^{window} = \begin{cases} Thread^{window}_{standard} + \left(B_{loan} - B_{boundary}\right) & if\ Condition1 \\ B_{request} & if\ Condition2 \\ B_{request} & if\ Condition3 \\ Thread^{window}_{standard} - \left(B_{boundary} - B_{loan}\right) & if\ Condition4 \\ B_{request} & if\ Condition5 \end{cases} \tag{8}$$

Attached:

$$\begin{aligned} &\textbf{\textit{Condition 1}}: B_{loan} > B_{boundary}\ \textbf{\textit{and}} \\ &\qquad B_{request} > Thread^{window}_{standard}\ \textbf{\textit{and}} \\ &\qquad Thread^{window}_{standard} + (B_{loan} - B_{boundary}) < B_{request} \\ &\textbf{\textit{Condition 2}}: B_{loan} > B_{boundary}\ \textbf{\textit{and}} \\ &\qquad B_{request} > Thread^{window}_{standard}\ \textbf{\textit{and}} \\ &\qquad Thread^{window}_{standard} + (B_{loan} - B_{boundary}) \geq B_{request} \\ &\textbf{\textit{Condition 3}}: B_{loan} > B_{boundary}\ \textbf{\textit{and}} \\ &\qquad B_{request} \leq Thread^{window}_{standard} \\ &\textbf{\textit{Condition 4}}: B_{loan} \leq B_{boundary}\ \textbf{\textit{and}} \\ &\qquad B_{request} \geq Thread^{window}_{standard} - \left(B_{boundary} - B_{loan}\right) \\ &\textbf{\textit{Condition 5}}: B_{loan} \leq B_{boundary}\ \textbf{\textit{and}} \\ &\qquad B_{request} < Thread^{window}_{standard} - \left(B_{boundary} - B_{loan}\right) \end{aligned}$$

The bandwidth allocation of the last thread in the polling cycle is mainly to compensate the prepaid bandwidth used by the previous thread. Take Condition 2 as an example to describe the scenario, and the remaining decision scenarios are similar. In Condition 2, the pre-payable bandwidth is greater than the initial prepayment limit. This indicates that the available bandwidth of the previous threads is greater than the prepaid bandwidth. Therefore, the resources available to this thread are the sum of the standard bandwidth and the bandwidth that exceeds the prepayment limit. When the sum of the two is greater than the thread demand, the thread can be directly allocated its required bandwidth.

C. BANDWIDTH ALLOCATION

After determining the allocated bandwidth of each thread, the bandwidth will be allocated to ONU according to the bandwidth request weight ratio of the ONU. According to the bandwidth request value, each ONU will get its weight ratio:

$$W_i = \frac{B^r_i}{\sum_{x=1}^{N} B^r_x} \tag{9}$$

where B_i^r represents the bandwidth request of onu_i, and $\sum_{x=1}^{N} B_x^r$ represents the total bandwidth requirement of all ONUs in the current thread.

According to the bandwidth window size of the thread where the ONU is located and the weight ratio of the ONU, each ONU can obtain a pre-allocated bandwidth:

$$B_i^{pre} = Thread^{window} * W_i \tag{10}$$

According to the relationship between pre-allocated bandwidth and requested bandwidth, the final allocated bandwidth of each ONU can be calculated by:

$$B_i^{end} = \begin{cases} B_i^r & if\ B_i^r > B_i^{pre}\ and\ B_i^r \le B_i^{pre} + B^{pool} \\ B_i^{pre} + B^{pool} & if\ B_i^r > B_i^{pre}\ and\ B_i^r > B_i^{pre} + B^{pool} \\ B_i^r & if\ B_i^r \le B_i^{pre} \end{cases} \tag{11}$$

where B^{pool} represents the bandwidth resource pool for the current thread.

In the process of allocating bandwidth to ONUs according to (11), when the requested bandwidth of the ONU is less than the pre-allocated bandwidth, the extra pre-allocated bandwidth will enter the bandwidth resource pool to participate in the subsequent ONU bandwidth allocation.

III. SLOT SCHEDULING OF MT-MTDC ALGORITHM

After allocating the bandwidth of ONUs, time slot scheduling needs to be performed for each thread's ONU. Through research, it can be known that in the single-threaded polling algorithm, the shortest propagation delay (SPD) first scheduling strategy can compensate for the wide propagation distance up to 100 km, but cannot achieve load balancing. The longest-first first-available (LFFA) [14] strategy can achieve excellent load balance during time slot scheduling, so as to effectively reduce the length of the polling cycle and improve bandwidth utilization. But for the MT-MTDC algorithm based on the multi-threaded polling mechanism, the LFFA strategy cannot meet the demand. Compared with the single-thread algorithm, the multi-thread algorithm has multiple threads in one polling cycle, and subsequent threads will allocate the wavelength slot resources based on the previous thread.

In an ideal situation (without considering tuning delay and conflict), the ONU scheduling of multi-threaded polling algorithm based on the LFFA strategy in a polling cycle is shown in Fig. 1. In Fig. 1, a polling cycle contains two threads, each ONU needs to be transmitted twice. There are two points to note here. First, the working wavelength of the ONU will be adjusted based on load balancing in the two transmissions. So, the ONU wavelength tuning is more frequent in the multi-threaded polling algorithm. In a polling cycle, the more threads, the greater the pressure on ONU scheduling. Second, there is a possibility of ONU transmission conflict in the multi-threaded polling algorithm. It can be seen from Fig. 1 that for ONU7, there is a conflict between the two transmissions on the time axis. The transmission of ONU7 in

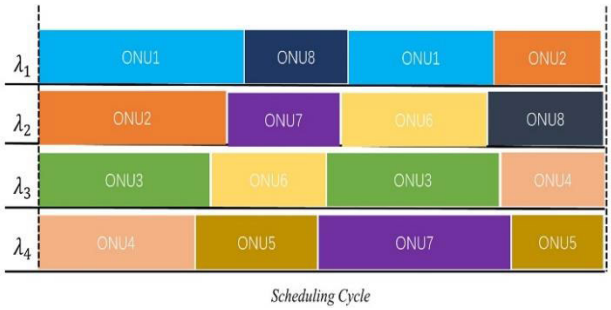

FIGURE 1. Example of ONU scheduling for multi-threaded polling algorithm.

the first thread is at wavelength λ_2, and in the second thread is at wavelength λ_4. The two transmissions have overlapping parts on the time axis. In the scenario where ONU tuning time is considered, wavelength tuning of ONU7 before the end of the first thread's transmission will affect the integrity of data transmission. Even if the tuning time of ONU is not considered, when ONU has only one transceiver, ONU7 cannot transmit simultaneously on two wavelengths at the same time. These are two main problems that need to be solved when LFFA strategy is applied to the multi-threaded polling algorithm.

A. BUILDING ONU TUNING BUFFER

In order to ensure wavelength load balancing, the working wavelength of the ONU is not fixed during the time slot scheduling process, and the ONU wavelength tuning needs to be considered in each thread. Therefore, the ONU tuning buffer will be built at the beginning of each thread. Some ONUs are selected for direct transmission without wavelength tuning, and the rest ONUs can use this period of time for wavelength tuning. While building ONU tuning buffer, it is still necessary to consider load balancing, so as to ensure that the overall bandwidth utilization rate is high.

First, arrange all ONUs in descending order of bandwidth and start traversing from the head of the queue. Then it is determined whether the operating wavelength of the ONU is the earliest idle wavelength, if yes, it is arranged for priority transmission, otherwise this ONU is skipped. The length of the tuning buffer also needs to be limited. And a global variable needs to be set to record the maximum tuning time of the ONU that is not currently allocated a time slot. When the buffer length of a certain wavelength can meet the tuning requirements of all subsequent ONUs, the buffer construction of that wavelength is completed. Every time the ONU slot allocation is determined, the global variable $tuning_{max}$ will be updated. At the same time, traverse from the head of the ONU queue again, until the buffers of all wavelengths are constructed. The detailed pseudo-code of the MT-MTDC algorithm for building ONU buffer is shown in Table 3.

TABLE 3. Build a buffer of the appropriate length for ONU wavelength tuning.

Algorithm 2: Build a buffer of the appropriate length for ONU wavelength tuning

Input: $wave\_count[\,]$, $onu\_wave\_stamp[\,][\,]$, $onu\_time\_stamp[\,][\,]$, $onu\_pool[\,][\,]$, $waves[\,][\,]$, n_{wave}^{active}, $tuning_{max}$
Output: $waves[\,][\,]$, $onu\_pool[\,][\,]$
1: function BuildBuffer($wave\_count[\,]$, $onu\_wave\_stamp[\,][\,]$, $onu\_time\_stamp[\,][\,]$, $onu\_pool[\,][\,]$, $wave[\,][\,]$, n_{wave}^{active}, $tuning_{max}$**)**
2: let $tuning_{max}$ be assigned the maximum tuning time of all ONUs in $onu\_pool[\,][\,]$
3: for i=0 → $n_{wave}^{active} - 1$ **do**
4: **if** ($wave\_count[i] < tuning_{max}$) **do**
5: **for** j=0 → $onu\_pool\_size - 1$ **do**
6: **if** $onu\_wave\_stamp[j] == i$ **and** $onu\_time\_stamp[j] \le wave\_count[i]$ **and** $wave\_count[i] < tuning_{max}$ **do**
7: $wave\_count[i] \mathrel{+}= onu\_pool[j][0]$
8: $onu\_time\_stamp[j] \mathrel{+}= onu\_pool[j][0]$
9: **fill** $onu\_pool[j]$ into the corresponding position in $waves[\,][\,]$
10: *pool.remove(j)*
11: *j = j - 1*
12: **update** the value of $tuning_{max}$
13: **else if** $wave\_count[i] \ge tuning_{max}$ **do**
14: **break**
15: return $waves[\,][\,]$, $onu\_pool[\,][\,]$
16: end function

TABLE 4. Allocate time slot to ONUs in the resource pool.

Algorithm 3: Allocate time slot to ONUs in the resource pool

Input: $wave\_count[\,]$, $onu\_wave\_stamp[\,][\,]$, $onu\_time\_stamp[\,][\,]$, $onu\_pool[\,][\,]$, $waves[\,][\,]$, $wave\_order[\,]$
Output: $waves[\,][\,]$
1: function Allocate($wave\_count[\,]$, $onu\_wave\_stamp[\,][\,]$, $onu\_time\_stamp[\,][\,]$, $onu\_pool[\,][\,]$, $waves[\,][\,]$, $wave\_order[\,]$**)**
2: let $wave\_order[\,]$ be stored the available order of all active waves
3: while (! *onu_pool.isEmpty()*) **do**
4: **i = 0**
5: **for** j=0 → $wave\_order\_size - 1$ **do**
6: **if** $onu\_time\_stamp[j] \le wave\_count[i]$ **do**
7: $wave\_count[i] \mathrel{+}= onu\_pool[j][0]$
8: $onu\_time\_stamp[j] \mathrel{+}= onu\_pool[j][0]$
9: **fill** $onu\_pool[j]$ into the corresponding position in $waves[\,][\,]$
10: $onu\_wave\_stamp[j] = wave\_order[i]$
11: **update** $wave\_order[\,]$
12: *onu_pool.remove(i)*
13: **break**
14: return $wave[\,][\,]$, $onu\_pool[\,][\,]$
15: end function

B. ONU TIME SLOT ALLOCATION

After the ONU tuning buffer of each wavelength is constructed, time slots will be allocated for ONU based on load balancing strategy, and the possible transmission conflicts will be corrected.

First, the ONUs that are not allocated time slots in the resource pool are organized into a descending queue according to the allocated bandwidth. Then, start traversing from the head of the queue. Since the construction of ONU buffer is completed, the influence of tuning time can be ignored. Assign the ONU to the first idle wavelength in sequence. To correct the transmission conflict, a time flag is set for each ONU in OLT. After determining the transmission time slot of each ONU on a thread, the flag is updated to the final transmission end time of the ONU. In the subsequent allocation process, the time flag of the ONU is compared with the planned transmission time of this thread. If there is a conflict, the ONU is temporarily skipped. If there is no conflict, the ONU is allocated a time slot. In each time, the time slot of an ONU is determined, it will return to the head of the queue to traverse again until all ONUs time slots scheduling end. The detailed pseudo-code of MT-MTDC algorithm for ONU time slot allocation is shown in Table 4.

The construction of the ONU buffer and the allocation of time slots containing transmission conflict corrections have effectively solved two problems: 1) The problem of more frequent ONU tuning delay during the multi-threaded polling process is solved. And it is suitable for the scenario where ONUs with different tuning time coexist. 2) The ONU transmission conflict is eliminated, and the data loss caused by the sudden interruption of ONU upstream transmission is avoided. At the same time, it can ensure that all transmission wavelengths have good load balances, so as to reduce the transmission cycle and improve the bandwidth utilization.

C. MT-MTDC ALGORITHM

The overall flow of the MT-MTDC algorithm is presented in Fig. 2.

As shown in Fig. 2, the MT-MTDC algorithm is divided into three steps:

Step1: Parameters determination. First, by an adaptive thread number and wavelength number selection mechanism, the number of threads and the number of wavelengths enabled is determined. Then, by dynamic adjustment, the size of the thread window is determined.

Step2: Bandwidth allocation. Firstly, the weight of ONU is calculated, and then the bandwidth of ONU is pre allocated according to W_i. Finally, the final allocated bandwidth is determined according to the relationship between the pre allocated bandwidth and the requested bandwidth.

Step3: Slot scheduling. First, construct the ONU tuning buffer. Then, the ONU time slot allocation is performed and the potential transmission conflict is corrected at the same time. It is worth mentioning that the above two processes are based on load balancing.

IV. SIMULATION AND PERFORMANCE ANALYSIS

In this section, the performance of the proposed MT-MTDC algorithm is studied. The simulation is developed in MATLAB and Java. By simulation, the MT-MTDC algorithm is compared with MT-LFFA algorithm and multi-tuning-time ONU scheduling (MOS) algorithm [15]. The MT-LFFA algorithm is LFFA algorithm [14] based on

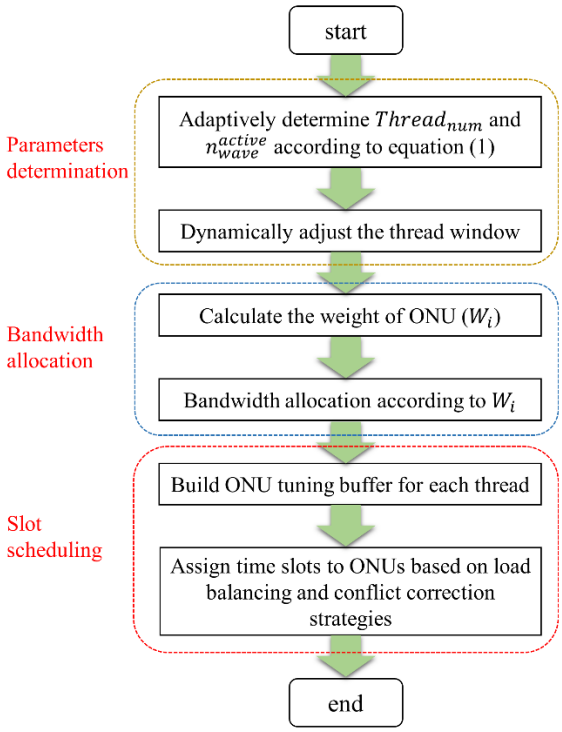

FIGURE 2. The flow chart of MT-MTDC algorithms.

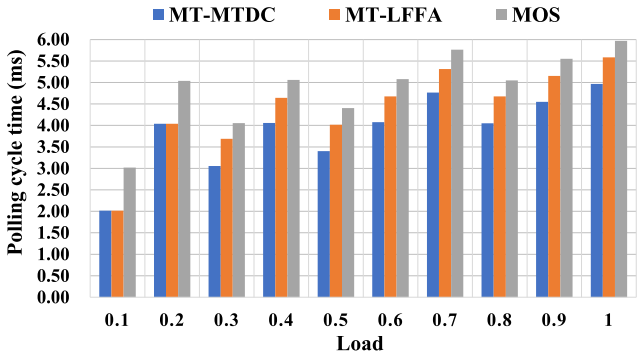

FIGURE 3. Comparison of polling cycle time of three algorithms.

multi-threaded polling. The MOS algorithm is the algorithm that based on single-threaded polling with different tuning time ONU coexistence. The simulation is focused on four aspects: polling cycle time, average tuning delay, bandwidth utilization and average packet delay.

In the simulation, there are four available wavelengths in the VPON based on LR WDM / TDM PON. VPON contains 128 ONUs, and the initial wavelength of each ONU are randomly assigned. The distance between ONU and OLT is evenly distributed within 0-100km. Studies have pointed out that the performance of multi-threaded algorithms will be affected by the polling cycle size [2]. In 2012, Ahmed Helmy *et al.* proposed that the optimal polling cycle for access networks with a maximum coverage of 100 km is 5 ms [4]. In this paper, the maximum polling cycle is 5ms. The data packet of ONUs is generated by Poisson distribution, and packet size ranges from 64 bytes to 1518 bytes (The unit will be converted to bits during the operation process) [15]. The guaranteed bandwidth of ONU is determined by the number of data packets it generates. The allocated guaranteed bandwidth of each packet is 64 bytes. Unless otherwise specified, the ONU tuning time in the simulation scenario is randomly selected from {0.1 ms, 0.3 ms, 0.5 ms}. The simulation data is the arithmetic average of 100 consecutive polling cycles.

A. ANALYSIS OF POLLING CYCLE TIME

With the same load and the same number of wavelengths enabled, the longer the polling cycle is, the more delay or idle slots exist in the transmission. In this section, the polling cycle of MT-MTDC algorithm, MT-LFFA algorithm and MOS algorithm at each load point will be compared. The comparison results are shown in Fig. 3. It can be seen from Fig. 3 that when the load points are 0.1 and 0.2, the polling cycle time of MT-MTDC algorithm and MT-LFFA algorithm is the same, while the polling cycle time of MOS algorithm is higher than that of the other two algorithms. This is because when the load points are 0.1 and 0.2, the three algorithms only enable one wavelength. There is no effect of tuning delay currently. Since the MOS algorithm is a single-threaded algorithm, the idle time slot caused by high RTT increases the polling cycle time. When the load points are 0.3, 0.5 and 0.8, the polling cycle time of the three algorithms decreases compared with the previous load. This is because at these three load points, the number of enabled wavelengths of the three algorithms increases. According to the load balancing mechanism, part of the transmission time slots will be allocated to the newly added wavelengths, resulting in a decrease in the polling cycle time. From an overall perspective, the polling cycle time of the MT-MTDC algorithm is the shortest, and the polling cycle time of the MOS algorithm is the longest. The polling cycle time of MT-LFFA algorithm is longer than that of MT-MTDC algorithm. This is because the ONU wavelength tuning and transmission conflict leads to waiting delay. MOS algorithm avoids the influence of ONU wavelength tuning through slot scheduling, and there is no transmission conflicts in single thread algorithm. However, in the VPON scenario based on LR WDM/TDM PON, the single-threaded algorithm is inevitably affected by high RTT, which will greatly increase the polling cycle time. From Fig. 3, it can be found that the polling cycle time of MT-LFFA algorithm and MOS algorithm at multiple load points are greater than the set maximum value of 5ms. For the scenario where the maximum polling cycle time is limited, the transmission data will be lost or delayed in the actual transmission process, which has a great impact on network service quality.

B. ANALYSIS OF AVERAGE TUNING DELAY

The tuning delay in this section refers to the waiting delay of ONU transmission queue caused by the fact that some ONUs cannot be transmitted within a given slot due to the

algorithm's failure to consider the tuning delay not to be zero or the coexistence of multiple tuning-time devices in the scheduling process.

The average tuning delay comparison of the three algorithms is shown in Fig.4. When the load points are 0.1 and 0.2, the number of enabled wavelengths of the three algorithms is 1. There is no tuning requirement for ONU currently. Therefore, the load interval of Fig. 4(a) is selected as [0.3, 1]. Detailed simulation data of all load points can be seen in Fig. 4(b). It can be seen from the line chart that when the load is in the range of [0.3, 1], the average tuning delay of the MT-MTDC algorithm and the MOS algorithm is always 0, while the MT-LFFA algorithm has a significant tuning delay. This is because both the MT-MTDC algorithm and the MOS algorithm consider the situation that the ONU tuning time is not zero and multiple tuning devices coexist, while the MT-LFFA algorithm does not. When performing bandwidth allocation, both the MT-MTDC algorithm and the MOS algorithm take the ONU tuning time into account, so the ONU can be transmitted in a given time slot during wavelength tuning. According to our definition of the tuning time, the ONU tuning delay of the MT-MTDC algorithm and the MOS algorithm is 0. The average tuning delay of the MT-LFFA algorithm fluctuates up and down at 0.6 ms, which has no obvious correlation with load. The average tuning delay of the MT-LFFA algorithm can be up to 0.64 ms. This is because both MT-LFFA algorithm and MT-MTDC algorithm have at least two threads in a polling cycle, and each thread may have a requirement for ONU wavelength tuning. The final tuning delay is the delay superposition of multiple threads. It is also proved that the effect of tuning on the delay of the multi-threaded algorithm is higher than that of the single-threaded algorithm.

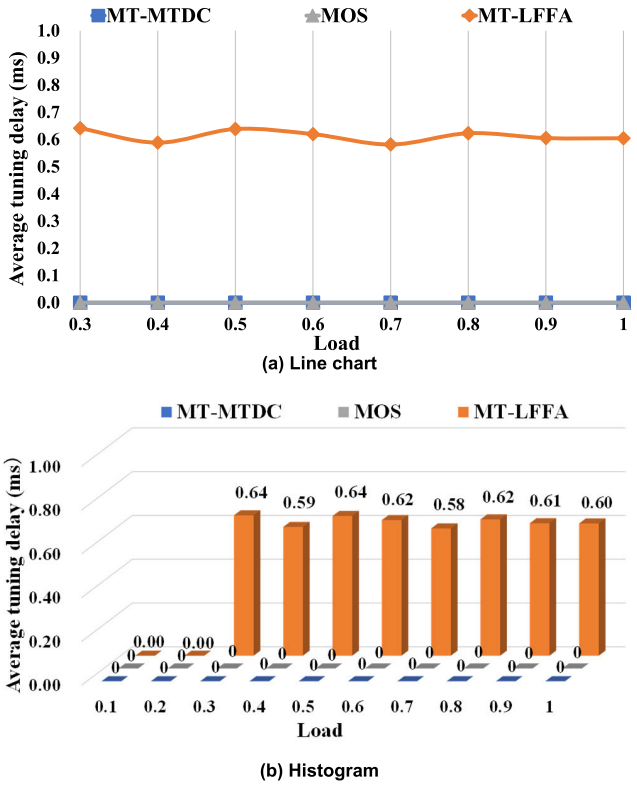

FIGURE 4. Comparison of average tuning delay of three algorithms: (a) Line chart; (b) Histogram.

C. ANALYSIS OF AVERAGE TUNING DELAY IN DIFFERENT SCENARIOS

The MT-MTDC algorithm is suitable for scenarios where devices with different tuning times coexist. Therefore, in this section, three different scenarios will be set to analyze the performance of MT-MTDC algorithm. The MT-LFFA algorithm, which is also a multi-threaded polling algorithm, will participate in the comparison. The tuning time parameters of device for specific scenarios are shown in Table 5.

TABLE 5. ONU device tuning parameters in three scenarios.

Scenario	Tuning time set of ONU device
A	{0.5ms, 0.75ms, 1ms}
B	{1ms, 1.5ms, 2ms}
C	{1ms, 1.25ms, 1.5ms}

In scenarios A, B, and C, the tuning time of all ONU devices is randomly selected from the parameter set. In the simulation, both MT-MTDC algorithm and MT-LFFA algorithm will continuously run 1000 polling cycles at each load point of each scenario. And the average tuning delay data of 1000 cycles will be compared and analyzed.

The average tuning delay of the two algorithms in scenario A is shown in Fig. 5. The load interval of Fig. 5(a) is [0.3, 1]. It is used to show the fluctuation of average tuning delay at different load points. Detailed simulation data of all load points can be seen in Fig. 5(b). As can be seen from Fig. 5(a), when the load is in the range of [0.3, 1], the MT-LFFA algorithm has a significant wavelength tuning delay during the upstream transmission, and the average tuning delay floats around 1.4 ms. The average tuning delay of the MT-MTDC algorithm is always kept at zero. This proves the effectiveness of dynamically selecting the number of threads. Comparing Fig. 4 and Fig. 5, it can be found that the average tuning delay is related to the ONU tuning time. The larger the ONU tuning time, the longer the tuning delay.

The average tuning delay of the two algorithms in scenario B is shown in Fig. 6. The load interval of Fig. 6(a) is [0.3, 1]. It is used to show the fluctuation of average tuning delay at different load points. Detailed simulation data of all load points can be seen in Fig. 6(b). The tuning time set of ONU in scenario B is {1 ms, 1.5 ms, 2 ms}. In scenario B, the ONU tuning delay in the upstream transmission of MT-LFFA algorithm can reach up to 3.6 ms, and the MT-MTDC algorithm also has a tuning delay. According to our simulation data, in 1000 consecutive polling cycles, the MT-MTDC algorithm has tuning delay in some polling cycles at all load points in the interval of [0.3, 1]. This indicates that

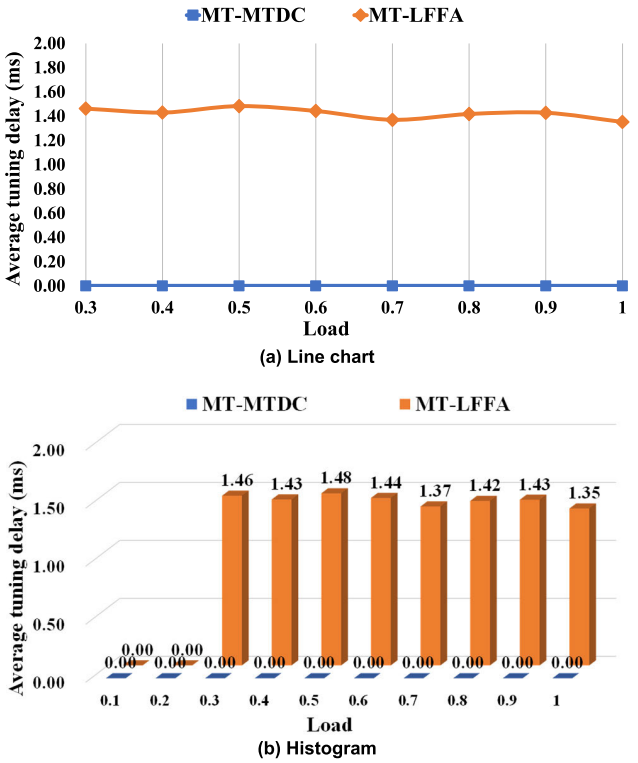

FIGURE 5. Comparison of the average tuning delay of the two algorithms in scenario A: (a) Line chart; (b) Histogram.

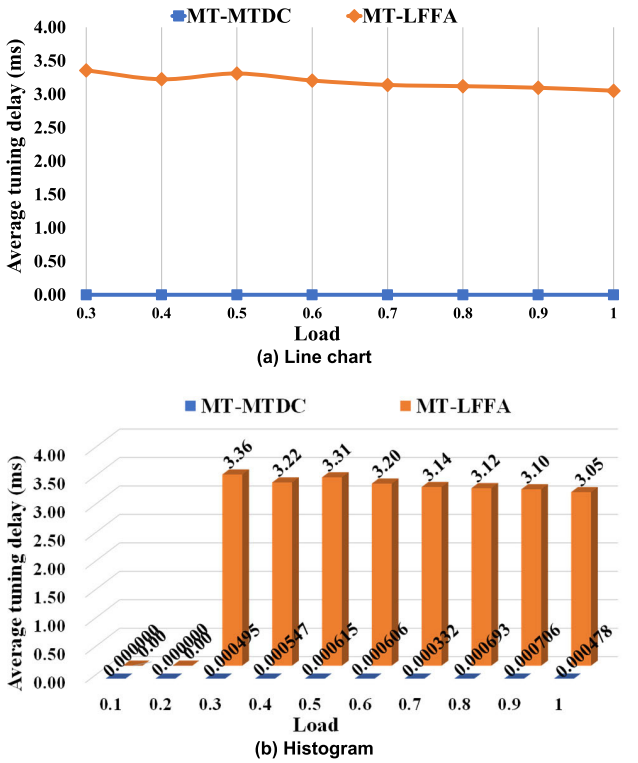

FIGURE 6. Comparison of the average tuning delay of the two algorithms in scenario B: (a) Line chart; (b) Histogram.

the ONU tuning buffer may not be long enough when the proportion of ONU tuning time in polling cycle is too high. The construction of the ONU tuning buffer depends on the load balancing in the last polling cycle and the fluctuation of the ONU bandwidth request in this polling cycle. However, from the specific data of each load point in Fig. 6(b), the average tuning delay of MT-MTDC algorithm in 1000 consecutive polling cycles is less than 1μs, which is completely negligible compared with the polling cycle of 5 ms and the delay of MT-LFFA algorithm up to about 3 ms. The maximum average tuning delay of the MT-LFFA algorithm is 3.36 ms, which means that each thread needs to provide a buffer time of about 1.7 ms through scheduling in the polling cycle. For the simulation environment, the standard time window length of a polling cycle is only 2.5 ms. Therefore, when the tuning time of ONU device is too high, MT-LFFA algorithm is not suitable for dynamic bandwidth allocation scenarios.

The average tuning delay of the two algorithms in scenario C is shown in Fig. 7. The load interval of Fig. 7(a) is [0.3, 1]. It is used to show the fluctuation of average tuning delay at different load points. Detailed simulation data of all load points can be seen in Fig. 7(b). The tuning time set of ONU in scenario C is {1 ms, 1.25 ms, 1.5 ms}. Compared with scenario A and scenario B, the ONU tuning time of scenario C is between them. Therefore, the average tuning delay of the MT-LFFA algorithm in scenario C is also between scenario A and scenario B, which fully proves that the average tuning time increases with the increase of ONU tuning time. For the simulation data of scenario C, it is necessary to point out that the MT-MTDC algorithm generates tuning delay in a few polling cycles out of 1000 polling cycles. Because the total number of samples is too small and the tuning time is too short, the statistical average tuning delay of MT-MTDC algorithm in Fig. 7(b) is still zero. However, according to the simulation data, when the MT-MTDC algorithm is at load points 0.3 and 0.5, the probability of generating tuning delay is higher than that of other load points, and no tuning delay is detected at the load points 0.7 and 1. Combined with Fig. 3, it can be found that when the load points are 0.3 and 0.5, the polling cycle time of MT-MTDC algorithm is lower than that of other load points, and when the load points are 0.7 and 1, the polling cycle time of the algorithm is the highest. Therefore, it can be inferred that when the tuning delay of ONU device is constant, the longer the polling period is and the smaller the proportion of tuning time is, the larger the length of ONU tuning buffer can be constructed, so it is less likely to generate tuning delay.

D. ANALYSIS OF BANDWIDTH UTILIZATION

Bandwidth utilization refers to the ratio of the bandwidth transmitted by VPON in a polling cycle to the total bandwidth occupied. The comparison of bandwidth utilization of the three algorithms at different load points is shown in Fig. 8. It can be seen from the figure that the bandwidth utilization

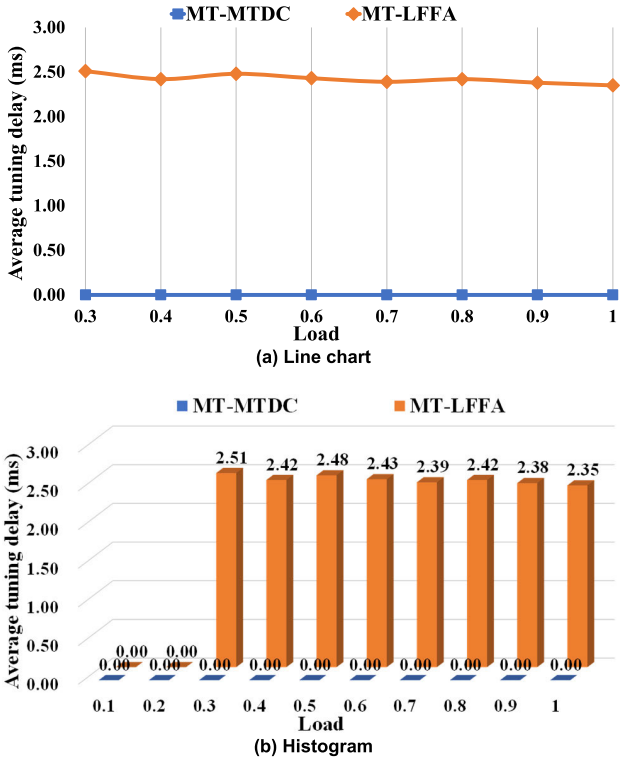

FIGURE 7. Comparison of the average tuning delay of the two algorithms in scenario C: (a) Line chart; (b) Histogram.

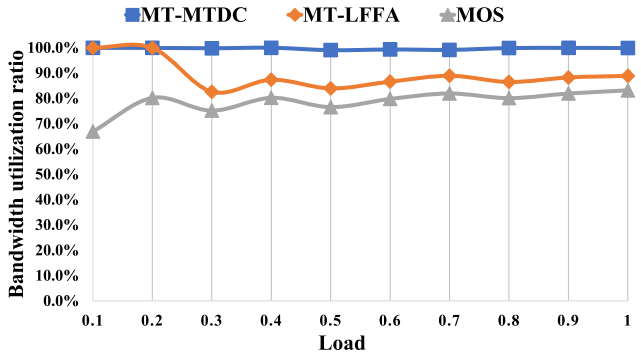

FIGURE 8. Comparison of bandwidth utilization of three algorithms.

of MT-MTDC algorithm is maintained around 99.9% all the time, and the part that does not reach 100% is the loss caused by the protection slot bandwidth(In order to reduce the impact of the delay caused by optical switching and the delay jitter of information transmission, protection slots need to be set between data packets in the process of ONU data transmission. The protection slot bandwidth is the sum of the bandwidth occupied by these protection slots.). When the load is greater than 0.3, the MT-MTDC algorithm is in a multi-wavelength state, and the bandwidth utilization remains at around 99%. This is because the MT-MTDC algorithm uses multiple threads to execute DBA, and schedules ONU according to load balancing strategy. Multi-threaded processing makes full use of the idle time slots caused by high RTT, greatly reduces the data packets delay. And load balancing can avoid the impact of data bursts. Therefore, higher bandwidth utilization is guaranteed.

For the MT-LFFA algorithm, when the load points are 0.1 and 0.2, the bandwidth utilization of the MT-LFFA algorithm is the same as that of the MT-MTDC algorithm. At this time, it is single wavelength operation, and there is no effect of load balance and ONU wavelength tuning. When the load is in the range of [0.3, 1], the bandwidth utilization of the MT-LFFA algorithm has obviously decreased. The reason for this decrease is the delay caused by the ONU wavelength tuning and the waiting delay caused by the transmission conflict of ONU. Since VPON supports the coexistence of ONU devices with different tuning times, the greater the tuning time of ONU devices, the greater the impact on bandwidth utilization.

In the whole range of [0.1, 1], the bandwidth utilization of MOS algorithm is much lower than that of the other two algorithms. This is because high RTT will result in the generation of idle slots. MOS algorithm is a single-threaded algorithm. When it works in the VPON scenario based on LR WDM/TDM PON, it needs to wait a long time after the transmission of the current polling cycle to receive the transmission scheduling of the next polling cycle, resulting in low bandwidth utilization. As the load increases, the RTT time remains unchanged, and the proportion of idle time slots gradually decreases. So, the bandwidth utilization of MOS algorithm will gradually increase with the increase of load in the four load intervals of [0.1, 0.2], [0.3, 0.4], [0.5, 0.7], [0.8, 1]. At load points of 0.3, 0.5 and 0.8, the bandwidth utilization of MOS algorithm decreases obviously. This is because the number of working wavelengths of MOS algorithm has changed at these load points. According to the characteristics of load balancing, with the increase of the number of wavelengths, the proportion of RTT idle time slots on each wavelength will increase. As a result, bandwidth utilization will decrease compared with the previous load point.

E. ANALYSIS OF AVERAGE PACKET DELAY

The average packet delay in this paper refers to the total time from packet generation to packet arrival at OLT. The average packet delay comparison of the three algorithms is presented in Fig. 9.

It can be seen from the figure that when the load is between 0.1 and 0.2, the average packet delay of the three algorithms is about 0.5 ms. This is because the waiting time of data transmission and the tuning time of ONU are very small at this time, and the average packet delay is maintained at half of RTT. When the load is greater than 0.2, the average packet delay of the three algorithms gradually increases. When the load is between 0.2 and 0.5, the MT-MTDC algorithm increases slowly compared with the MT-LFFA algorithm and the MOS algorithm. Because the multi-threaded processing of the MT-MTDC algorithm makes the ONU's waiting time

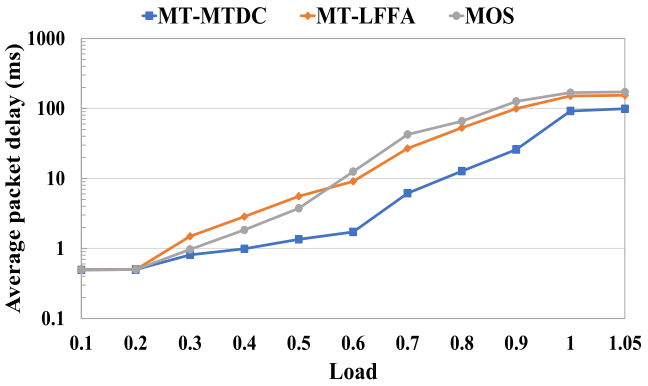

FIGURE 9. Comparison of average packet delay of three algorithms.

very short, the average packet delay of the MT-MTDC is also very small. However, due to the tuning delay, the average packet delay of MT-LFFA algorithm is the largest among the three algorithms. When the load is between 0.6 and 1, the average packet delay of the MOS algorithm is the largest. This is because the MOS algorithm is a single-threaded algorithm. When the load is heavy, the waiting time of the data packet of the single-threaded algorithm is longer than that of the multi-threaded algorithm. Since the MT-LFFA algorithm has no collision detection mechanism, the average packet delay is also relatively large when the load is heavy. When the load is greater than 1, the average packet delay of the three algorithms tends to be stable, and there is no big fluctuation. Overall, since the simulation is implemented in a long-distance scenario, the average packet delay of the three algorithms is not very small. Compared with the MT-LFFA algorithm and the MOS algorithm, the MT-MTDC algorithm has the best performance in reducing average packet delay.

V. CONCLUSION

In this paper, a MT-MTDC bandwidth allocation algorithm has been proposed. This algorithm solved the problem of high RTT and ONU tuning delay in VPON based on LR WDM/TDM PON. By adaptively selecting the number of wavelengths and threads, high bandwidth utilization is guaranteed. By the flexible thread window and bandwidth prepayment mechanism, the cooperation between threads is strengthened, and the degradation problem of the multi-threaded algorithm is solved. By constructing tuning buffer and setting time flag, the problems of more frequent ONU tuning pressure and ONU transmission conflict caused by multi-threaded polling mechanism are effectively solved. By simulation, the effectiveness of the proposed algorithm is demonstrated.

REFERENCES

[1] J. Navarro-Ortiz, P. Romero-Diaz, S. Sendra, P. Ameigeiras, J. J. Ramos-Munoz, and J. M. Lopez-Soler, "A survey on 5G usage scenarios and traffic models," *IEEE Commun. Surveys Tuts.*, vol. 22, no. 2, pp. 905–929, 2nd Quart., 2020, doi: 10.1109/COMST.2020.2971781.

[2] S. Saha, M. Hossen, and M. Hanawa, "A new DBA algorithm for reducing delay and solving the over-granting problem of long reach PON," *Opt. Switching Netw.*, vol. 31, pp. 62–71, Jan. 2019.

[3] H. Song, B.-W. Kim, and B. Mukherjee, "Multi-thread polling: A dynamic bandwidth distribution scheme in long-reach PON," *IEEE J. Sel. Areas Commun.*, vol. 27, no. 2, pp. 134–142, Feb. 2009, doi: 10.1109/JSAC.2009.090205.

[4] A. Helmy, H. Fathallah, and H. Mouftah, "Interleaved polling versus multi-thread polling for bandwidth allocation in long-reach PONs," *J. Opt. Commun. Netw.*, vol. 4, no. 3, pp. 210–218, 2012.

[5] A. Mercian, M. P. McGarry, and M. Reisslein, "Offline and online multi-thread polling in long-reach PONs: A critical evaluation," *J. Lightw. Technol.*, vol. 31, no. 12, pp. 2018–2028, Jun. 2013, doi: 10.1109/JLT.2013.2262766.

[6] A. Dixit, G. Das, B. Lannoo, D. Colle, M. Pickavet, and P. Demeester, "Adaptive multi-gate polling with void filling for long-reach passive optical networks," in *Proc. 13th Int. Conf. Transparent Opt. Netw.*, Jun. 2011, pp. 1–4, doi: 10.1109/ICTON.2011.5970943.

[7] M. De Andrade, A. Buttaboni, M. Tornatore, P. Boffi, P. Martelli, and A. Pattavina, "Optimization of long-reach TDM/WDM passive optical networks," *Opt. Switching Netw.*, vol. 16, pp. 36–45, Apr. 2015.

[8] A. Buttaboni, M. De Andrade, and M. Tornatore, "A multi-threaded dynamic bandwidth and wavelength allocation scheme with void filling for long reach WDM/TDM PONs," *J. Lightw. Technol.*, vol. 31, no. 8, pp. 1149–1157, Apr. 2013.

[9] A. Buttaboni, M. De Andrade, and M. Tornatore, "Dynamic bandwidth allocation with void filling and multi-thread for long reach WDM/TDM PONs," in *Proc. 15th Int. Telecommun. Netw. Strategy Planning Symp. (NETWORKS)*, Rome, Italy, 2012, pp. 1–6, doi: 10.1109/NETWKS.2012.6381716.

[10] H. Feng, C.-J. Chae, A. V. Tran, and A. Nirmalathas, "Cost-effective introduction and energy-efficient operation of long-reach WDM/TDM PON systems," *J. Lightw. Technol.*, vol. 29, no. 21, pp. 3135–3143, Nov. 2011, doi: 10.1109/JLT.2011.2166109.

[11] W. Xia, C. Gan, S. Ma, L. Xu, and W. Xie, "Multi-channel scheduling algorithm for multi-subsystem-based VPON in metro-access optical network," *Opt. Switching Netw.*, vol. 21, pp. 58–66, Jul. 2016.

[12] W. Xie, N. Zhan, C. Gan, Y. Yan, and H. Qiao, "(Step-variable 0-1 knapsack)-based mode transformation algorithm and universal control mechanism in a multisubsystem-based virtual passive optical network," *Opt. Eng.*, vol. 56, no. 3, Mar. 2017, Art. no. 036109.

[13] J. Zhang and N. Ansari, "Scheduling hybrid WDM/TDM passive optical networks with nonzero laser tuning time," *IEEE/ACM Trans. Netw.*, vol. 19, no. 4, pp. 1014–1027, Aug. 2011, doi: 10.1109/TNET.2010.2093150.

[14] Y. Zhang, C. Gan, K. Gou, and J. Hua, "(Box-filling-model)-based ONU schedule algorithm and bandwidth-requirement-based ONU transfer mechanism for multi-subsystem-based VPONs' management in metro-access optical network," *Opt. Fiber Technol.*, vol. 36, pp. 10–18, Jul. 2017.

[15] X. Wang, C. Gan, and L. Tong, "Adaptive scheduling algorithm for the coexistence of ONUs with different tuning time in virtual passive optical network," *IEEE Photon. J.*, vol. 11, no. 5, pp. 1–8, Oct. 2019, doi: 10.1109/JPHOT.2019.2944274.

[16] R. A. Butt, M. W. Ashraf, M. Faheem, and S. M. Idrus, "A survey of dynamic bandwidth assignment schemes for TDM-based passive optical network," *J. Opt. Commun.*, vol. 41, no. 3, pp. 279–293, 2018.

[17] K. A. Memon, K. H. Mohammadani, A. A. Laghari, R. Yadav, B. Das, W. U. Khan Tareen, N. U. A. Memon, and X. Xin, "Dynamic bandwidth allocation algorithm with demand forecasting mechanism for bandwidth allocations in 10-gigabit-capable passive optical network," *Optik*, vol. 183, pp. 1032–1042, Apr. 2019.

[18] L. Zhang, J. Qi, K. Wei, W. Zhang, Y. Feng, and W. Hou, "High-priority first dynamic wavelength and bandwidth allocation algorithm in TWDM-PON," *Opt. Fiber Technol.*, vol. 48, pp. 165–172, Mar. 2019.

[19] R. A. Butt, S. M. Idrus, S.-U. Rehman, P. M. A. Shah, and N. Zulkifli, "Comprehensive polling and scheduling mechanism for long reach gigabit passive optical network," *J. Opt. Commun.*, vol. 40, no. 1, pp. 1–12, 2019.

[20] Y. Senoo, "Hitless λ-tuning sequence to reduce tuning delay in λ-tunable WDM/TDM-PON," *J. Opt. Commun. Netw.*, vol. 8, no. 7, pp. 486–494, Jul. 2016, doi: 10.1364/JOCN.8.000486.

[21] A. Buttaboni, M. De Andrade, M. Tornatore, and A. Pattavina, "Dynamic bandwidth and wavelength allocation with coexisting transceiver technology in WDM/TDM PONs," *Opt. Switching Netw.*, vol. 21, pp. 31–42, Jul. 2016.

LIJUAN WU received the B.S. degree in communication engineering from Shanghai University, Shanghai, China, in 2019, where she is currently pursuing the M.S. degree in communication and information with the School of Communication and Information Engineering. Her research interests include future optical access network architecture and dynamic bandwidth allocation in virtual passive optical networks.

CHAOQIN GAN received the B.S. degree in physics from Nanchang Normal University, Nanchang, China, in 1990, and the M.S. degree in electronics engineering and the Ph.D. degree in communication engineering from Southeast University, Nanjing, China, in 1998 and 2001, respectively. In 2001, he joined Alcatel Shanghai Bell Company, Ltd. (Alcatel-Lucent) as a Senior Engineer in optical communications. Since 2007, he has been a Professor with the School of Communication and Information Engineering, Shanghai University, China. He has authored or coauthored more than 140 papers published in journals and conferences and holds 25 invention patents. His research interests include broadband optical access networks, multi-wavelength optical networking, and high-speed optical communication systems. He is a member of the Expert Group of the Shanghai Innovation Foundation and a Senior Member of the Chinese Institute of Electronics. From 2001 to 2005, he was a member of the Expert Group of Shanghai Optical Science and Technology.

XINGDI WANG received the B.S. and M.S. degrees in communication engineering from Shanghai University, China, in 2017 and 2020, respectively. He currently works at Huawei Technologies Company, Ltd. His research interest is dynamic bandwidth allocation in broadband optical access networks.

ZHONGSEN XU received the B.S. degree in communication engineering from Anhui Architecture University, Anhui, China, in 2018. He is currently pursuing the M.S. degree in communication and information with the School of Communication and Information Engineering, Shanghai University, Shanghai, China. His research interest is dynamic bandwidth allocation in broadband optical access networks.

JIANQIANG HUI received the B.S. degree in communication engineering from Southwest University, Chongqing, China, in 2018, where he is currently pursuing the M.S. degree in electronic and communication engineering with the School of Communication and Information Engineering. His research interest includes the novel architecture of WDM/TDM passive optical networks.