A High Speed Scheduler With a Novel Scheduling Algorithm for Optical Burst Switching Networks

Tairan Zhang, Guiling Wu, Xinwan Li, and Jianping Chen

***Abstract*—In this paper, a Max-CU-VF (Maximum Channel Utilization with Void Filling) channel scheduling algorithm for OBS (Optical Burst Switching) is proposed, which chooses the feasible data channel with the maximum channel utilization as the optimal one. The channel utilization defined in Max-CU-VF is the total length of scheduled BDPs (Burst Data Packet) on a data channel in a limited observing time window. The scheduling time can be decreased due to the elimination of time consumed in searching procedure in comparison with the traditional schedulers based on LAUC-VF (Latest Available Unused Channel with Void Filling). The hardware processing scheme and the corresponding hardware architecture for Max-CU-VF are designed in detail. A 16-channel Max-CU-VF based scheduler is demonstrated on FPGA (Field Programmable Gate Array). A scheduling time of 12.5 ns per BCP (Burst Control Packet) is achieved on the real-time running Max-CU-VF based scheduler. The performance of the Max-CU-VF is also compared with LAUC-VF through NS2 simulation. The results show that the burst loss ratio and the average throughput of Max-CU-VF are close to those of LAUC-VF when the load is light, and the faster Max-CU-VF outperforms the slower LAUC-VF very much when the arrival rate of bursts is over the processing speed of LAUC-VF. Therefore, Max-CU-VF is more suitable for future high speed OBS networks.**

***Index Terms*—Channel scheduling, FPGA, Max-CU-VF, OBS.**

I. Introduction

FOR the reason of combining the advantages and avoiding the shortcomings of OCS (Optical Circuit Switching) and OPS (Optical Packet Switching), the OBS (Optical Burst Switching) has been regarded as a promising solution to support IP over DWDM for next-generation optical switching networks [1], [2]. In OBS, the BCP (Burst Control Packet) is ahead of its corresponding BDP (Burst Data Packet) and processed electronically after O/E at switching nodes to reserve the valid wavelength for its corresponding BDP based on the information stored in it. As a consequence, data transparency is achieved by switching BDP all-optically at burst level.

Manuscript received April 11, 2012; revised July 24, 2012, October 05, 2012, and May 04, 2013; accepted June 18, 2013. Date of publication June 26, 2013; date of current version August 06, 2013. This work was supported in part by 973 project (2011CB301700), NSFC (61071011, 61127016), ISTCP (2011DFA11780), the STCSM Project (10DJ1400402, 12XD1406400) and the project of State Key Lab of Advance Optical Communication Systems and Networks at Shanghai Jiao Tong University, China. *(Corresponding author: G. Wu).*

The authors are with the State Key Laboratory of Advanced Optical Communication Systems and Networks, Shanghai Jiao Tong University, Shanghai 200240, China (e-mail: ztr1918294@163.com; wuguiling@sjtu.edu.cn; lixinwan@sjtu.edu.cn; jpchen62@sjtu.edu.cn).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/JLT.2013.2271237

In the ultra-high speed OBS networks, the arrival rate of BCPs may be over the service rate of scheduler if its processing speed is not fast enough, which makes the input queue become longer and longer. It will cause some BCPs to be discarded due to limited storage memory or insufficient offset time. For avoiding this kind of burst loss, a high speed scheduler is needed. For example, for a system with 80 wavelengths per link and the bandwidth of each wavelength is 100 Gbps, we need to process a BCP every 16 ns when the average size of BDPs is 16 KB. With the explosive growth of the cloud computation, the mass data of real-time video or online game and so on, the scheduling speed must be promoted to keep up with the higher and higher optical transmission rate. At the same time, for the same input traffic at the ingress node, the average length of burst can be shorter when the scheduler could process more bursts in unit time. Not only can shorter bursts reduce the assembling time at ingress nodes, but also improve the flexibility and channel utilization of the network [3]. As a result, the speed of the channel scheduling becomes very important and the algorithms with complicated procedure or slow processing speed are not practical for future high speed OBS networks.

The LAUC (Latest Available Unused Channel) [4] and Horizon [5] are less complicated and seized of the speed advantage since they only need to examine the end time of the last scheduled BDP on each data channel. Their bandwidth utilizations, however, are lower. A rescheduling algorithm with $O(c)$ is shown in [6], where c is the number of channels. But it brings "ghost bursts" which will increase the overhead of networks and bring unnecessary collisions.

FF-VF (First Fit with Void Filling) can reach higher speed through eliminating the procedure to find the optimal channel among feasible channels by taking the first found feasible channel as the optimal one [7]. However, the elimination of the optimal channel selection leads to poorer bandwidth utilization. Moreover, further improvement of processing speed of FF-VF is limited since it still needs to search the feasible channel among all data channels, which takes up most of scheduling time. M. T. Anan *et al.* ever proposed a SWAP (Slotted Wavelength Assignment Pipeline) scheduler to implement an algorithm similar to FF-VF with a high speed using a simple digital logic circuit [8]. However, the SWAP scheduler cannot utilize scheduling slots efficiently since it uses the fixed slot's duration and the FF-VF algorithm.

Rather than processing BCPs in order of their arrival time one by one, the CTBR (Constant Time Burst Resequencing) scheduler in [9] delays and resequences a batch of BCPs, and then processes them according to corresponding BDPs arrival time. After that, the scheduler could use simpler horizon algorithm to achieve $O(1)$ operation on FPGA. However, that kind of scheduling scheme needs complicated resequencing implementation

of BCPs and extra FDLs (Fiber Delay Line) that range from a few microseconds to a few tens of microseconds at the input ports of each OBS node. Besides, CTBR is not adapted to the offset-based priority scheme.

LAUC-VF [7] has good channel utilization since it tries to utilize the idle voids among scheduled BDPs adequately. However, the complexity of a straight forward implementation of LAUC-VF is $O(m)$, where m is the number of the voids in all data channels. In order to achieve as good channel utilization as LAUC-VF and high processing speed at the same time, several channel scheduling mechanisms have been proposed, such as Min-SV [10] and an index-based scheduler [11]. Min-SV decreases the computation complexity of LAUC-VF from $O(m)$ to $O(log(m))$ by adopting a set of geometry-based algorithms to implement LAUC-VF. The index-based scheduler implements LAUC-VF using an index-based searching approach to find feasible channels on different channels in parallel to reach $O(1)$ time complexity.

However, since Min-SV and the index-based scheduler still needs to take some time to select out the feasible channels, it is very hard to promote its processing speed further to satisfy the requirement of future ultra-high speed optical networks. In this paper, we consider to promote the processing speed of scheduler by adopting a scheduling algorithm with a new metric to rapidly select out optimal channel and designing suitable hardware architecture for it at the same time. In order to speed up the processing, the corresponding information used by the new algorithm to select out the optimal data channel should be obtained in a simple way instead of the complex searching procedure used by index-based scheduler. Here, we define a CU (Channel Utilization), the total length of scheduled BDPs on a data channel in a limited observing time window, as the metric of the optimal data channel selection and present a Max-CU-VF based scheduler accordingly, which chooses the data channel with the maximum defined CU value as the optimal one from several feasible data channels. The performance of Max-CU-VF is evaluated and compared with LAUC-VF through NS2 simulation. Moreover, a scheduler based on the Max-CU-VF is implemented on FPGA. The results show that the proposed scheduler can achieve higher speed than the index-based scheduler whereas its performances of burst loss ratio and average throughput are nearly the same or much better than those of LAUC-VF when the arrival rate of bursts is less or greater than the service capability of index-based scheduler for LAUC-VF, respectively.

The rest of this paper is organized as follows. Section II describes the Max-CU-VF scheduling algorithm and related working scheme in detail. The hardware architecture and implementation results are shown in Section III. A 16-channel Max-CU-VF based scheduler is also demonstrated in the section. Section IV presents the results of comparison between LAUC-VF and Max-CU-VF in terms of burst loss ratio and average throughput through NS2 simulation. Finally, Section V concludes the paper.

II. The Max-CU-VF Scheduling Algorithm and Related Working Scheme

Nearly all of the void filling channel scheduling schemes for OBS can be divided into two phases. The first phase is to identify all the data channels with suitable voids which can accommodate the incoming BDP. The second phase is to choose the optimal one from feasible data channels found in the first phase according to some kinds of metrics. Searching for feasible data channels is the main procedure in scheduling, which takes up most of the scheduling time.

Actually, the processing speed of the first phase is related to the metric of the optimal channel selection used in the second phase. Therefore, LAUC-VF has to experience a complicated searching process to get the accurate time information needed by the metric for optimal channel selection, which makes its complexity be $O(m)$. Although Min-SV and index-based scheduler have promoted the scheduling speed by simplifying the searching process or using faster searching approach, due to adopting the same metric to find optimal channel, they still have to run the searching process to get the accurate time information of voids needed in the second phase. Thus, we consider designing a faster scheduling algorithm using a new metric to select out the optimal channel without that searching process, and at the same time, making it have as good channel utilization as that of LAUC-VF.

It is well known that LAUC-VF achieves higher channel utilization by trying its best to minimize the void before the newly arriving burst. Since LAUC-VF enhances channel utilization through compressing the space of voids as much as possible, we can directly take the channel utilization as the metric to achieve the same effect. From this point of view, we use CU as the criteria to find out optimal channel and propose a Max-CU-VF algorithm which sets the coming burst on one of the feasible channels with maximum channel utilization. Because Max-CU-VF prefers to arrange the bursts on the channels with high utilization, other channels that contain the voids with longer average size may be left as much as possible for subsequent bursts. After a period of time, from the statistical point of view, the bandwidth of all channels are used efficiently as much as possible, just like the case happening in LAUC-VF. Meanwhile, the computation complexity of Max-CU-VF can be decreased since Max-CU-VF changes the selection metric from detailed time information of voids (micro-level) to channel utilization (macro-level), which neglects some unnecessary and superfluous information without impairing its channel utilization.

However, it is not practical to obtain the accurate bandwidth utilization of each channel in the operation of networks. Considering the channel selection is only dependent on relative CU values among different channels, we propose to define the CU of a channel as the total length of scheduled BDPs on the channel in an observing time window (the length of window is fixed), which is easy to be calculated through simple operation. Accordingly, a Max-CU-VF channel scheduling scheme is proposed. It selects out the optimal channel with the maximum defined CU value from feasible data channels. Because the calculation of CU on each channel has nothing to do with the time information of special voids around new BDP according to above definition, the complicated searching procedure used by LAUC-VF based schedulers can be avoided in Max-CU-VF based schedulers. Moreover, the similar bitwise operation approach used in SWAP can be applied in the implementation of Max-CU-VF to promote the scheduling speed significantly.

$$\begin{aligned} N &> (T_{off\_Max} + T_{BDP\_Max})/\tau \\ &> (T_{off\_Max} + T_{BDP\_Max})/T_{BDP\_Min} \end{aligned} \quad (1)$$

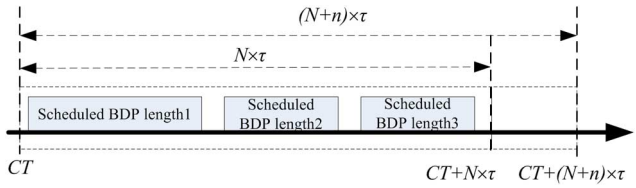

Fig. 1. The illustration of the calculation of CU.

In Max-CU-VF, the time window is divided into N slots with fixed length equally. According to [11], to avoid data access collisions, the slot size τ should be less than T_{BDP_Min} and N must satisfy (1). $T_{off-Max}$ is the maximum offset time. $T_{BDP-Max}$ and $T_{BDP-Min}$ are the time lengths of longest and shortest BDPs.

Considering the time before a scheduling has elapsed and it cannot be used by coming bursts, we set the start of each scheduling (called current time) as the beginning of the time window. In that case, when the size of the time window satisfies (1), all the BDPs after the current time are included in the time window. It means the value of CU will not change with the range of the time window when the size of the window satisfies (1). For example, in Fig. 1, the CU in the longer time window $[CT, CT + (N + n) \times \tau]$ is the same as the CU in $[CT, CT+N\times\tau]$ (N and τ satisfy (1)) since it is impossible for any BDP existing in the range of $[CT+N\times\tau, CT+(N+n)\times\tau]$ at current time CT.

Corresponding to the N time slots, there should be an N-bit binary register CH, an N-entry start time table, and an N-entry end time table on each data channel in the scheduler. A register CH is used to record the occupation state of the slots on a data channel. The Kth bit of CH will be set as logical 1 when part of scheduled BDP located in the Kth slot. Otherwise, the Kth bit is set as zero. The start and the end time of a scheduled BDP are recorded in the entries of the start and the end time tables corresponding to the slots where the head and the tail of the BDP located, respectively. Fig. 2 shows an example of the data structure. The corresponding value of each CH register and the time tables are also displayed in it. The detailed procedure of the Max-CU-VF based scheduler is composed of five steps.

$$Head = \left\lfloor \frac{(T_h - T_s)}{\tau} \right\rfloor \tag{2}$$

$$Tail = \left\lfloor \frac{(T_t - T_s)}{\tau} \right\rfloor \tag{3}$$

Step 1: Determine the index number of the slots where the head and tail of the new BDP are located in. They can be calculated through (2) and (3), where T_s is the start of the observing time window, T_h and T_t are the arrival and departure time of the new BDP, and $\lfloor\cdot\rfloor$ is a rounding down operation. In Fig. 2, the *Head* and *Tail* of the new BDP are 2 and 5.

Step 2: Generate the code to show which of slots will be covered by the new BDP through following (4). Where, $\ll$ is the left-shift operation and $|$ is the *bitwise OR* manipulation. For the new BDP in Fig. 2, $8'h01 \ll Tail$ results in 8'b00100000, and $8'h01 \ll Head$ is 8'b00000100. The *NewBDP* is 8'b00111100 after the operation, which means the newly coming BDP will set its foot from slot2 to slot5.

$$NewBDP = ((8'h01 \ll Tail) - (8'h01 \ll Head)) \,|\, (8'h01 \ll Tail) \tag{4}$$

Step 3: Find out feasible data channels by operating the *bitwise AND* between the *NewBDP* and the CH of each data channel in parallel. The result R is classified into five cases to find out all feasible data channels for the new BDP.

Case 1: Only one bit of R is logic 1 and the head of new BDP is just located in the corresponding slot, which means the head of the new BDP and the tail of a previously scheduled BDP are in the same slot. Contrasting to SWAP in [8], where the channel is simply judged as unfeasible one when the mentioned situation happens, the proposed scheduler directly picks up the end time of the scheduled BDP from the end time table and compares it with the start time of new BDP to make more accurate judgment for the object of utilizing the bandwidth better. If the end time of the scheduled BDP is less than the start time of new BDP, the channel is feasible. Otherwise, the channel is not suitable. For example, in Fig. 2, the $R1$ is 8'b00000100. Both the head of new BDP and the tail of a scheduled BDP1 are located in slot2 of the first data channel. Then the end time of scheduled BDP1 $(T1)$ is compared with the start time of new BDP $(H9)$. The first channel is not feasible for the new BDP since $H9 < T1$.

Case 2: Only one bit of the R is logic 1 and the tail of new BDP is located in the corresponding slot, which means the tail of the new BDP and the head of a scheduled BDP are in the same slot indicated by the logic 1 of R (see the second data channel in Fig. 2 and $R2$ is 8'b00100000). In this case, the start time of scheduled BDP is read out from the start time table and compared with the end time of new BDP to judge the data channel is feasible or not. In Fig. 2, the second channel is also unfeasible since $T9 > H2$.

Case 3: Two bits of R are logic 1, and the head and tail of new BDP are located in the two corresponding slots, respectively, which indicates the new BDP may collide with two previously scheduled BDPs at its head and/or tail part. The case is divided into two sub cases in terms of the number of slots occupied by the new BDP further.

When the new BDP occupies more than two slots, extract the end time of scheduled BDP before new BDP and the start time of the scheduled BDP after new BDP from related time tables of the two slots corresponding to the two logic 1 bits in R. After that, compare them with the start and the end time of the new BDP respectively to determine whether the channel is feasible or not for the new BDP. For example, in Fig. 2, two bits in $R3$ (8'b00100100) corresponds to slot2 and slot5. The end time of the scheduled BDP3 $(T3)$ in slot2 and the start time of the scheduled BDP4 $(H4)$ in slot5 are compared with the start time $(H9)$ and the end time $(T9)$ of the new BDP. Since $T3 < H9$ and $T9 < H4$, the third channel is feasible. So is the fourth channel.

On the other side, when new BDP takes up only two slots, seeing in Fig. 3, the R will have the same value (8'b00011000)

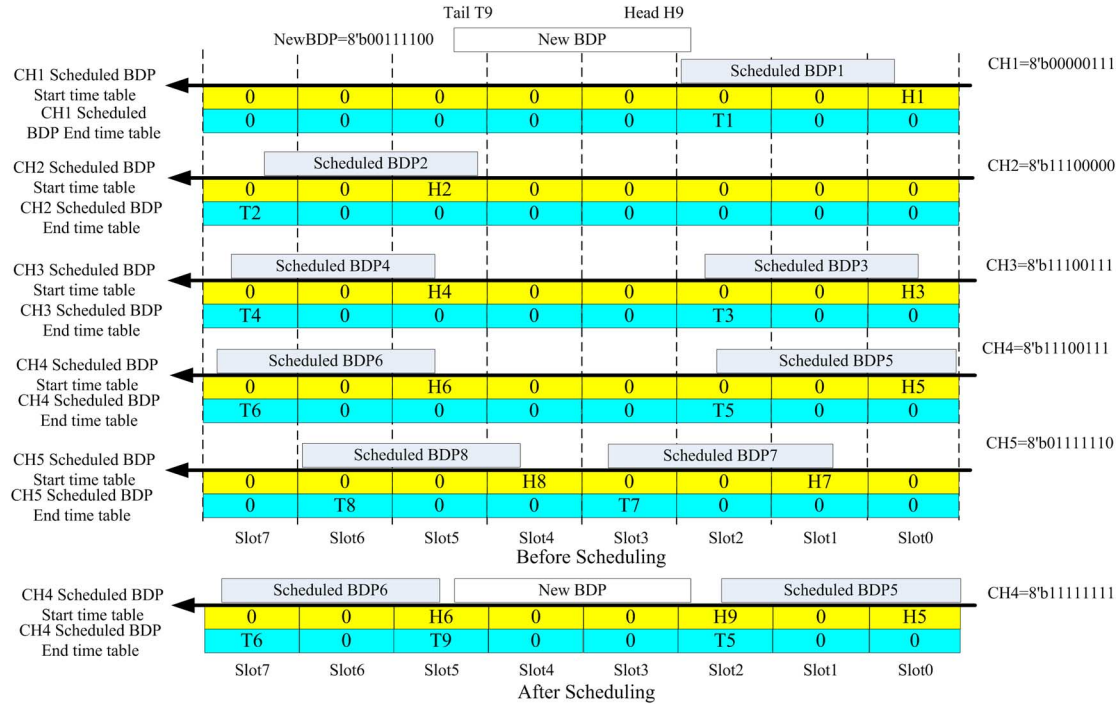

Fig. 2. The data structure of scheduling scheme when a new BDP takes up more than two slots.

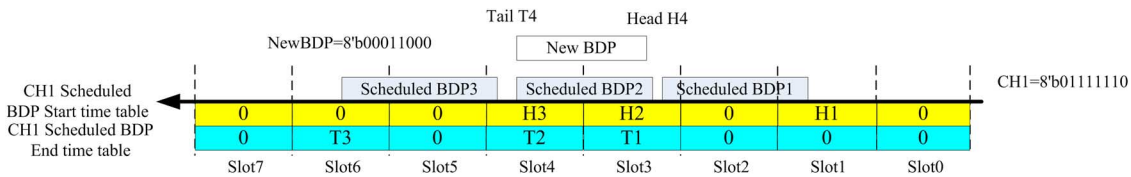

Fig. 3. The special case when a new BDP only occupies two slots.

no matter whether the scheduled BDP2 exists. The channel is feasible only when the values of slot3 in the start time table and slot4 in the end time table are zero with $T1 < H4$ and $T4 < H3$. Obviously, new BDP overlaps with scheduled BDP2.

Case 4: R is zero, which means the new BDP does not contend with any scheduled BDP on the data channel. The data channel is feasible obviously.

Case 5: R is the value except for above four cases. In this case, the data channel is unusable. The fifth data channel in Fig. 2 shows an example of the cases, where $R5$ is 8'b00111100.

After above process, all feasible channels (the third and fourth channels in Fig. 2) can be found out. We also observe the time slots, partially occupied by scheduled BDPs (slot2 and slot5 in both channel 3 and 4 in Fig. 2), can still be used by a new BDP since the feasible channels are only decided according to the start time and end time of voids and the new BDP in the proposed scheme, which makes the resource utility of the proposed scheme not related to the slot size τ and distinguishes it from other slot-based schemes such as SWAP in [8] where a time slot can only be occupied by one BDP.

Step 4: Select the optimal channel with maximum CU from all feasible channels. In Fig. 2, the fourth channel is selected as optimal channel according to Max-CU-VF whereas the third channel is considered as the optimal one under LAUC-VF.

Step 5: Update correlative information of optimal channel in parallel. The value of $CH4$ is refreshed through the *bitwise OR* operation between *NewBDP* and old value of $CH4$. In parallel, $H9$ and $T9$ are written into the start and the end time table separately, and the CU of the optimal channel will be updated by adding up the length of the new BDP. After the step, the scheduler goes back to the first step for next scheduling at once.

The procedures of index-based scheduler and Max-CU-VF based scheduler are listed in Table I. Their processing complexities are also compared in the table step by step. From the table, we can see that the step 1, 2, 4, and 5 of Max-CU-VF based schedulers correspond to the step 1, 2, 5, and 6 of index-based schedulers, respectively, and their complexities are almost same. In the third step, the index-based scheduler needs to locate the right time slot to get the metric value by a searching procedure, which takes up most of the scheduling time of index-based schedulers. Whereas the Max-CU-VF based scheduler completely eliminates the searching procedure by adopting a new metric CU which can be obtained directly from fixed registers. That is the main reason why the Max-CU-VF based scheduler is better in speed.

III. Hardware Implementation

Fig. 4 shows the hardware architecture of Max-CU-VF based scheduler on FPGA. The O/E receiver with burst-mode CDR (Clock Data Recovery) writes the recovered BCPs into the electronic input FIFO in order one by one. The central control unit

TABLE I
THE COMPLEXITY COMPARISON OF TWO SCHEDULERS

Index-based scheduler	Max-CU-VF based scheduler	Comparison of their complexities
1. Determine the slot where the head of a new BDP locates.	1. Determine the slot where the head and the end of a new BDP locate.	Their complexities are nearly the same since the Max-CU-VF based scheduler can locate slots in parallel.
2. Generate needed mask code for next step through bitwise logical operation.	2. Generate needed code for next step through bitwise logical operation.	Their complexities are almost the same since both of them use the same bitwise logical operation method.
3. Make bitwise *AND* operation, and search for candidate slot to get the time information needed by step 4	3. Make bitwise *AND* operation, and find the feasible channels by comparing the time information directly from registers.	The Max-CU-VF does not have the searching procedure and is better in speed.
4. Find the feasible channels through comparing the time information		
5. Find the optimal channel from the feasible channels by comparing their metric values	4. Find the optimal channel from the feasible channels by comparing their CUs	Their time complexities are same since their operation are same when the numbers of data channels are equal.
6. Update the information of the optimal channel	5. Update the information of the optimal channel	Different information can be updated in parallel, so Their time complexities are equivalent.

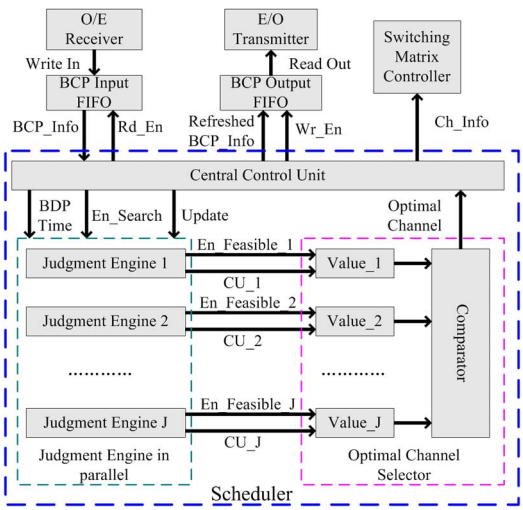

Fig. 4. The hardware architecture of Max-CU-VF based scheduler.

reads out the first BCP in the FIFO after a BCP scheduling is finished. At the same time, the control unit enables the judgment engine whose work is to select out all feasible data channels. The *En_Feasible_i* is high when the *i*th channel is feasible. *Value_i* is set to be *CU_i*, the channel utilization of the *i*th channel, when *En_Feasible_i* is high. Otherwise, *Value_i* is set to be zero. After that, all the values from *Value_1* to *Value_J* are sent to the comparator in parallel to acquire the maximum value quickly by using the method in [12]. Finally, the central control updates the corresponding information on optimal channel. Some fields of the processed BCP, like the offset time and channel ID and so on, are also refreshed. The updated BCP is written into the output FIFO, waiting for being transmitted out to the next switching node by the E/O transmitter. The switching matrix is configured by the control unit through the switching matrix controller according to the scheduling results.

There are two kinds of logic circuits for implementation on FPGA: the time sequencing logic circuit and the combinational logic circuit. Time sequencing logic circuit always finishes each step in one clock and can achieve much high clock frequency. The combinational logic circuit finishes all steps within a short time which can be regarded as the scale of one clock. A 16-channel scheduler based on Max-CU-VF is implemented on Xilinx Virtex4 XC4VFX20 FPGA using time sequencing logic circuits and combinational logic circuits, respectively. For comparison, parameters such as time window, BDP and offset time are set to the same ones used by the index-based scheduler in [11]. The length of each slot is 256 clock cycles. The size of the time window is 32 slots. The BDP duration has a uniform distribution between [256, 2560] clock cycles, and the offset time of each BDP is randomly taken between [1280, 5376] clock cycles with a uniform distribution.

TABLE II
THE COMPARISON OF THE PROCESSING TIME BETWEEN SCHEDULERS

Scheduler	Clock frequency	Number of clocks	Processing time of per BCP
Index-based scheduler using combinational logic circuit	50MHz	1	20ns
Max-CU-VF based scheduler using combinational logic circuit	80MHz	1	12.5ns
Index-based scheduler using time sequencing circuit	150MHz	6	40ns
Max-CU-VF based scheduler using time sequencing circuit	200MHz	5	25ns

The processing performance achieved by each scheduler is shown in Table II. The corresponding performance of the index-based scheduler is also listed in the table. From it, we can see that the processing time per BCP of the Max-CU-VF based schedulers is 32.5% less than that of the index-based schedulers no matter what kind of logic circuits is adopted, which is due to eliminating the searching procedure in the third step of index-based schedulers according to the analysis in Table I. The implementation reports from ISE (Xilinx Inc.) also show that the hardware resource consumed by the Max-CU-VF based scheduler is 6% less than the index-based one when adopting combinational logic circuits. We can also see that the processing time of the schedulers using the sequencing logic circuit is twice of that using the combinational circuit despite its higher clock frequency. The main reason is that the number of clocks consumed by the time sequencing logic circuit is much more than that consumed by the combinational logic circuit. It indicates the combinational logic circuit is more suitable to implement the high-speed scheduler on FPGA.

Fig. 5(a) is the circuit simulation result of the Max-CU-VF based scheduler using combinational logic circuits in the ModelSim (Mentor Inc.). The tested bursts were generated and put into the input FIFO of the scheduler in advance since our main aim is to validate the scheduling processing. We can see that the clock frequency of the scheduler reaches 80 MHz and

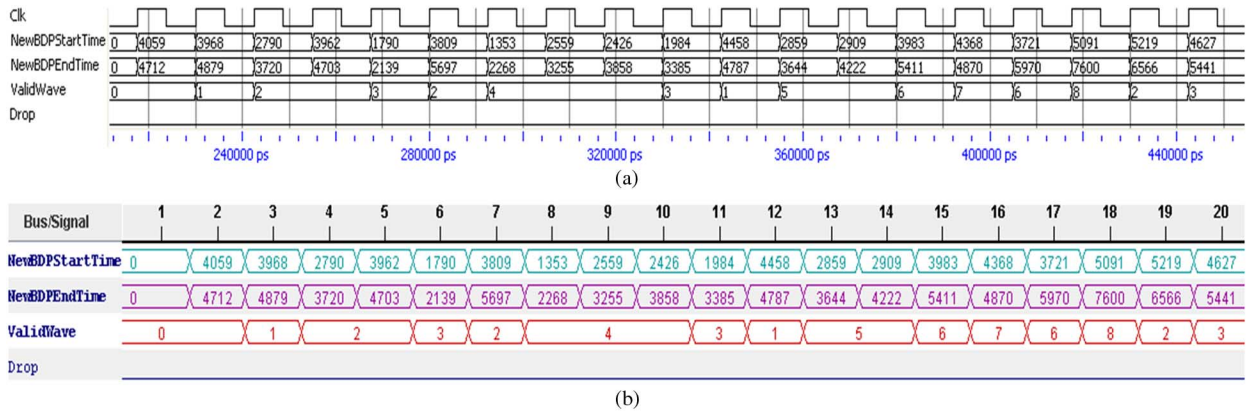

Fig. 5. The circuit simulation of scheduling process (a) and the real-time data captured from ChipsSope (b) of a 16-channel scheduler.

the processing time per BCP is only 12.5 ns accordingly. The *ValidWave* shows the index number of the optimal wavelength channel found by scheduler. The *ValidWave* is zero and the signal *Drop* is set to be high at the same time when the scheduler does not find feasible data channel for a new BDP. The start and the end time of each new BDP are one clock ahead of its corresponding *ValidWave* value. The Fig. 5(b) shows the real time data captured from running FPGA through ChipScope (Xilinx Inc.). We can see that the results in the two figures are consistent with each other, which proves the scheduler works very well on hardware.

IV. Performance Simulation

In this section, the performances of the Max-CU-VF are evaluated and compared with LAUC-VF by NS2 simulation in the NSFNet topology, which is shown in Fig. 6. We assume the bandwidth of each channel is 100 Gbps. Each core node has full wavelength conversion capability and no optical buffers. Each edge node (responsible for assembling/disassembling BDPs from/into original data) connects to a core node through a WDM link. The original packets arrive at each edge node in a *Poisson* distribution. The generated BDPs at each edge node are uniformly destined to all other 13 edge nodes and transmitted along the fixed OBS paths determined by the SPF (shortest path first) algorithm. The size of the time window is set to meet the minimum requirement of (1). The scheduling time in every core node is set as 20 ns and 12.5 ns for LAUC-VF and Max-CU-VF, respectively, according to the hardware implementation results in Section III. The load is defined as the ratio of total data arrival bit rates at all ingress nodes over the total bandwidth of output optical links of all ingress nodes. For example, if the number of data channels in a link is D, the total bandwidth of 14 output optical links is $14 \times D \times 100$ Gbps. Therefore, the load of the network is 1 when the total arriving bits at 14 ingress nodes is $14 \times D \times 100$ gigabits in one second.

Fig. 7 shows the burst loss ratios and average throughputs of LAUC-VF and Max-CU-VF when the BDP length is 12.5 KB and D is 8. In this case, the burst arrival rate will never exceed the service rate of LAUC-VF, which means all the burst loss is caused by no feasible channel for new coming burst. From the Fig. 7, we can see the loss ratio and average throughput of Max-CU-VF are both very close to those of LAUC-VF. The results prove the channel utilization of MAX-CU-VF is as good

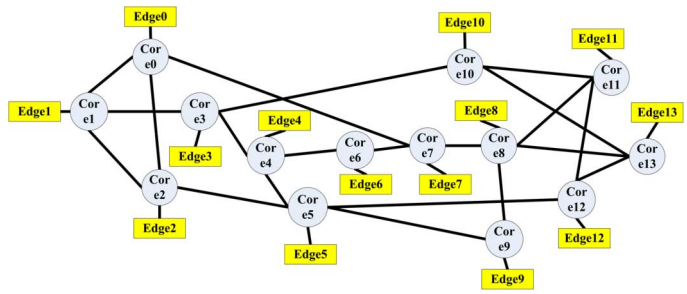

Fig. 6. The simulation network topology based on NSFNet.

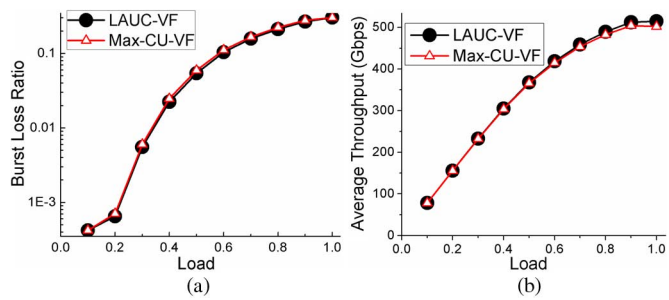

Fig. 7. The burst loss ratio (a), average throughput (b) when the BDP length is 12.5 KB and the number of data channels in a link is 8.

as that of LAUC-VF when the burst arrival rate is below the service rate of LAUC-VF.

Fig. 8 shows the burst loss ratios and average throughputs of LAUC-VF and Max-CU-VF when the BDP length is 14 KB and the D is 32. In the figure, LAUC-VF begins to lose bursts since the arrival rate of bursts starts to exceed the service rate of LAUC-VF when the load is 0.3813. In the range of 0.3813 to 0.4, with the load increasing, about 40% of the newly injected bursts are dropped for no enough offset time. That is because most of the bursts with the maximum hops (about 44% of all bursts) are likely to be dropped for lack of enough offset time. With the load increasing, the congestion problem caused by the slow scheduling speed of LAUC-VF becomes more and more serious. Then more and more bursts with fewer hops, besides the bursts with maximum hops, will be dropped for no sufficient offset time, which makes the burst loss ratio of LAUC-VF increasing sharply. However, for Max-CU-VF, the burst loss ratio is always much lower than that of LAUC-VF since it has higher

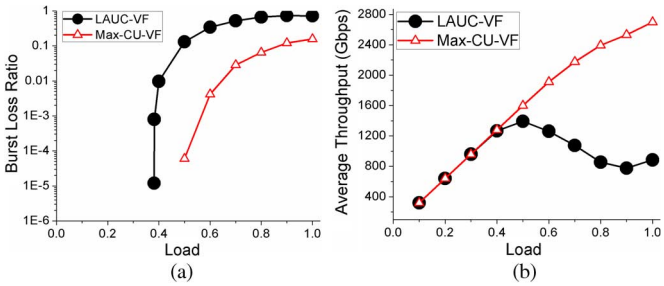

Fig. 8. The burst loss ratio (a), average throughput (b) when the BDP length is 14 KB and the number of data channels in a link is 32.

speed to process the coming bursts. When the load is larger than 0.6, the Max-CU-VF has to drop a few of bursts since the bursts arrival rate is a little over its processing capability, which makes a loss ratio of about 10% when the load is 1. In Fig. 8(b), the throughput of LAUC-VF reaches its maximum value when the load is about 0.5 and descends when the load is greater than 0.5 for the serious congestion at heavy load. The throughput of Max-CU-VF keeps quick rising before the load reaching 0.8 and its increasing slope becomes a little flatter later as its burst loss ratio is getting a little worse.

V. Conclusion

In this paper, we propose a Max-CU-VF channel scheduling algorithm for OBS, which uses the channel utilization in a limited time window as the criteria to select out the optimal channel. Because the defined channel utilization can be obtained without a complicated searching procedure, the presented Max-CU-VF based scheduler can execute faster than the schedulers based on LAUC-VF. The hardware architecture and the processing scheme for Max-CU-VF are presented in detail. A 16-channel Max-CU-VF based scheduler using combinational logic circuits is implemented on FPGA, which is able to complete one scheduling procedure every 12.5 ns. The performance of the Max-CU-VF is also evaluated and compared with LAUC-VF by NS2 simulation. The results show that Max-CU-VF is close to LAUC-VF in terms of the burst loss ratio and average throughput when the load is light, and outperforms LAUC-VF much better when the bursts arrival rate exceeds the processing speed of LAUC-VF.

Future works include the theoretical model and analysis of Max-CU-VF, the development of faster Max-CU-VF based scheduler which is able to process multiple BCPs in parallel.

References

[1] C. Qiao and M. Yoo, "Optical burst switching (OBS)-a new paradigm for an optical internet," *J. High Speed Netw.*, vol. 8, no. 1, pp. 69–84, 1999.

[2] X. Li, J. Chen, G. Wu, H. Wang, and A. Ye, "An experimental study of an optical burst switching network based on wavelength-selective optical switches," *IEEE Commun. Mag.*, vol. 43, no. 5, pp. 3–10, May 2005.

[3] J. Choi, H. Le Vu, C. W. Cameron, M. Zukerman, and M. Kand, "The effect of burst assembly on performance of optical burst switched networks," *Inf. Netw.*, vol. 3090, pp. 729–739, 2004.

[4] Y. Xiong, M. Vandenhoute, and H. Cankaya, "Design and analysis of optical burst-switched networks," in *Proc. SPIE'99 Conf. All Optical Networking: Architecture, Control, Management Issues*, Boston, MA, USA, Sep. 19–22, 1999, vol. 3843, pp. 112–119.

[5] J. S. Turner, "Terabit burst switching," *J. High Speed Netw.*, vol. 8, no. 1, pp. 3–16, 1999.

[6] S. K. Tan, G. Mohan, and K. C. Chua, "Algorithms for burst rescheduling in WDM optical burst switching networks," *Comput. Netw.*, vol. 41, no. 1, pp. 41–55, 2003.

[7] Y. Xiong, M. Vandenhoute, and H. C. Cankaya, "Control architecture in optical burst-switched WDM networks," *IEEE J. Select. Areas Commun.*, vol. 18, no. 10, pp. 1838–1851, Oct. 2000.

[8] M. T. Anan and G. M. Chaudhry, "A real-time hardware-based scheduler for next-generation optical burst switches," in *Proc. IEEE Int. Conf. Communication*, Glasgow, U.K., Jun. 2007, pp. 2289–2293.

[9] Y. Chen, J. S. Turner, and P. F. Mo, "Optimal burst scheduling in optical burst switched networks," *J. Lightw. Technol.*, vol. 25, no. 8, pp. 1883–1894, Aug. 2007.

[10] J. Xu, C. Qiao, J. Li, and G. Xu, "Efficient burst scheduling algorithms in optical burst-switched networks using geometric techniques," *IEEE J. Select. Areas Commun.*, vol. 22, no. 9, pp. 1796–1811, Sep. 2004.

[11] G. Wu, T. Zhang, J. Chen, X. Li, and C. Qiao, "An index-based parallel scheduler for optical burst switching networks," *J. Lightw. Technol.*, vol. 29, no. 18, pp. 2766–2773, Sep. 2011.

[12] F. Callegati, A. Campi, and W. Cerroni, "Fast and versatile scheduler design for optical packet/burst switching," *Opt. Switch. Netw.*, vol. 8, no. 2, pp. 93–102, 2010.

Tairan Zhang received the B.S. degree from China University of Mining and Technology, Xuzhou, China, in 2005 and M.S. degree from East China Normal University, Shanghai, China, in 2008, respectively. He is currently studying toward the Ph.D. degree in optical communication at the Shanghai Jiao Tong University, Shanghai, China. His research focuses on the modeling and experimental study of optical burst switching networks.

Guiling Wu received the B.S. degree from Haer Bing Institute of Technology, Heilongjiang, China, in 1995, and the M.S. and Ph.D. degrees in optical electrical engineering from Huazhong University of Science and Technology, Wuhan, China, in 1998 and 2001, respectively. He is currently an Associate Professor with the State Key Laboratory of Advanced Optical Communication Systems and Networks, Department of Electronic Engineering, Shanghai Jiao Tong University, Shanghai, China. His research interest includes optical networking and high-speed optical signal processing.

Xinwan Li received the M.S. degree from Shanghai University, Shanghai, China, in 1993, and Ph.D. degree from Shanghai Jiao Tong University, Shanghai, China, in 2005. Since 1993, he has been with Shanghai Jiao Tong University, Shanghai, China, where he is currently a Professor. From 1997 to 1998, he was with Essex University, U.K., as a Research Assistant. In 2001 he joined OPCOM Inc. as an Engineer and as a visiting professor of Chonbuk National University in 2007. His main research interest includes optical switching technologies and advanced optical fiber components. Prof. Li is a senior member of the IEEE Photonics Society and the chair of IEEE Communications Society Shanghai chapter.

Jianping Chen received the B.S. degree from Zhejiang University, Hangzhou, China, in 1983, and the M.S. and Ph.D. degrees from Shanghai Jiao Tong University, Shanghai, China, in 1986 and 1992, respectively. He is currently a Professor with the State Key Laboratory of Advanced Optical Communication Systems and Networks, Department of Electronic Engineering, Shanghai Jiao Tong University. His main research topics cover photonic devices and signal processing, optical networking, and sensing optics. He is also a principal scientist of 973 project in China.