Contour-Based Priority Scheduling in Optical Burst Switched Networks

Yuhua Chen, *Member, IEEE*, Jonathan S. Turner, *Fellow, IEEE*, and Zhi Zhai, *Student Member, IEEE*

***Abstract*—Optical burst switching (OBS) is an emerging technology that allows variable size data bursts to be transported directly over dense wavelength division multiplexing links. Although several quality-of-service (QoS) schemes have been proposed for OBS networks, how to provide QoS at the high speed required by the OBS network is still an open question. In this paper, we propose a novel O(1) runtime contour-based priority algorithm that provides complete priority isolation among different priorities. This is the first practical O(1) runtime priority algorithm proposed for OBS, and it is well suited to high-speed hardware implementation.**

***Index Terms*—Algorithm, optical burst switching (OBS), optical packet switching, scheduling, wavelength division multiplexing (WDM).**

I. Introduction

OPTICAL BURST switching (OBS) [1]–[4] is an emerging technology that allows variable size data bursts to be transported directly over dense wavelength division multiplexing (DWDM) links. In OBS networks, the control information is delivered on a separate control channel. Shortly before the transmission of data bursts, a burst header cell (BHC) is transmitted on the control channel, setting up and tearing down optical wavelength paths for data bursts on-the-fly. Data bursts can remain in the all-optical data plane and pass intermediate switching nodes transparently.

In an OBS network, packets with the same destinations are assembled together to form a burst. In order for the OBS to become a viable solution for the next-generation Internet, the OBS network must be able to provide quality-of-service (QoS) to various applications. In electronic routers, random access memories (RAMs) are extensively used to buffer packets in order to provide differentiated services. However, there is no equivalent RAM in the optical domain. The only way to provide a limited amount of delay is to use fiber delay lines (FDLs). As a result, the QoS schemes used in electronic routers cannot be applied to the OBS networks.

Although several QoS schemes have been proposed for OBS networks, the following question remains open: How do we design QoS schemes that can operate at the high speed required by the OBS networks? For example, for a 16-port system with 64 wavelengths per link, each operating at 10 Gb/s, we need to process one BHC every 78 ns in order to support an average burst length of 100 kBs. To the best of the authors' knowledge, this stringent requirement is beyond the capability of any proposed QoS schemes.

In this paper, we propose a novel O(1) runtime contour-based priority (CBP) algorithm. This is the first practical O(1) runtime algorithm that is proposed to support QoS in OBS networks.

The rest of this paper is organized as follows. Section II describes the OBS network architecture. Section III gives a detailed survey of existing QoS schemes. Section IV describes the proposed CBP algorithm, analyzes the properties that enable O(1) runtime operations, and discusses the hardware implementation. We conclude this paper in Section V.

II. OBS Network Architecture

The basic concept of an OBS network is illustrated in Fig. 1. The OBS network consists of a set of core routers and edge routers connected by DWDM links. In OBS networks, data bursts are assembled at ingress edge routers and disassembled at egress edge routers. For example, IP packets with the same destination edge router address are assembled into the same burst. Data bursts remain in the optical domain and pass through the core routers transparently.

In OBS networks, at least one DWDM channel per link is used as the control channel to send control information. The rest of the channels are data channels that carry data bursts. Before the transmission of a data burst, a BHC is sent on the control channel, specifying the offset time between the BHC and the burst, and the length (time duration) of the burst. Fig. 2 shows the timing relationship between the BHCs and their corresponding bursts. We use the term BHC to denote burst headers for convenience purposes. In practice, burst headers can take any format that is supported by the electronic control path in an OBS router.

BHCs on the control channel are converted to electronic signals and processed electronically by OBS core routers. The OBS core routers dynamically set up and tear down lightpaths based on the information carried in the BHCs. Optical data bursts can pass through the core routers without O/E/O conversion.

An OBS core router consists of an optical data path and an electronic control path, as shown in Fig. 3. The optical data path is the optical crossconnect with/without wavelength

Manuscript received November 2, 2006; revised May 2, 2007.

Y. Chen and Z. Zhai are with the Department of Electrical and Computer Engineering, the University of Houston, Houston, TX 77204-4005 USA (e-mail: yuhua.chen@mail.uh.edu; Zhi.Zhai@mail.uh.edu).

J. S. Turner is with the Department of Computer Science and Engineering, Washington University in St. Louis, St. Louis, MO 63130 USA (e-mail: jst@cse.wustl.edu).

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.2007.901332

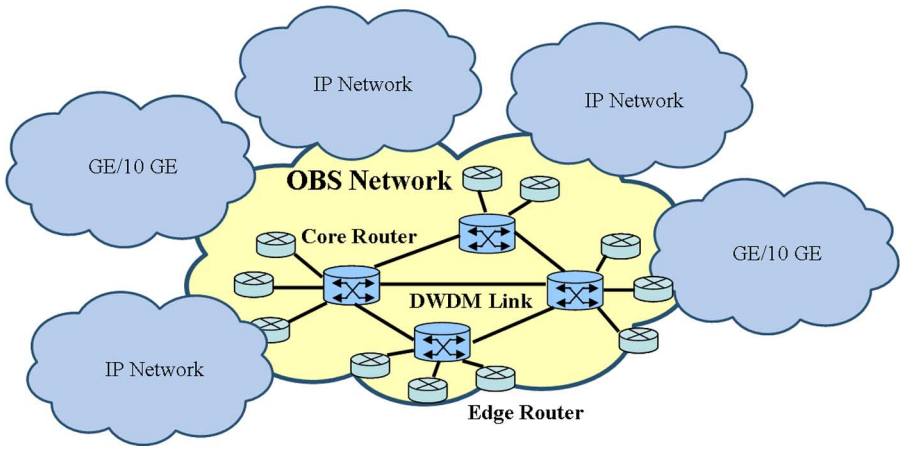

Fig. 1. OBS concept.

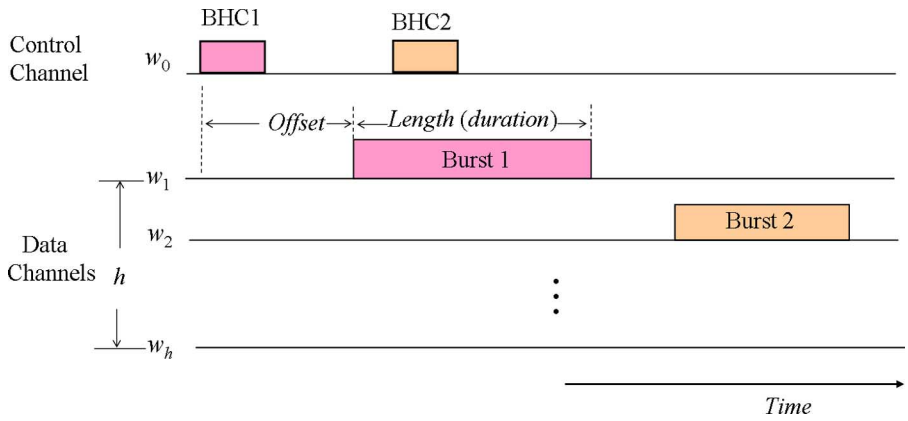

Fig. 2. Bursts and BHCs.

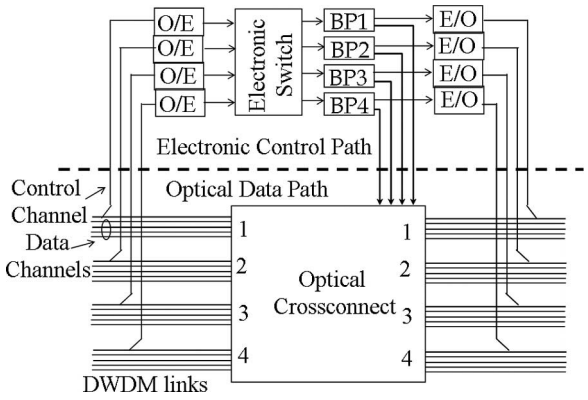

Fig. 3. OBS core router architecture.

conversion capability. The electronic control path contains O/E/O conversion, an electronic switch, and a set of BHC processing units, which we call burst processors (BPs). BPs process BHCs electronically, selecting outgoing wavelength channels for the arriving bursts. An optical path will be set up in the optical crossconnect before the arrival of the data burst so that the burst can pass through the optical interconnects without delaying or converting back to the electronic form. The CBP algorithm proposed in this paper is implemented electronically in the BPs.

III. Related Work

In this section, we provide a comprehensive summary of existing QoS schemes for OBS networks that are most relevant to this paper.

A. Offset-Based QoS

The offset-based QoS [5] was the first QoS scheme proposed for OBS networks. It assigns an extra offset time to bursts with higher priority. As a result, high-priority bursts are scheduled ahead of the low-priority bursts and have a better chance of reserving a wavelength successfully. The offset-based QoS can provide a complete priority isolation among different priorities. Unfortunately, it is well known that such a scheme has undesirable end-to-end delay, and it favors bursts with shorter lengths [6].

In addition, in order to support the offset-based QoS, wavelength scheduling algorithms with void-filling capability are needed. Unfortunately, such algorithms have higher computational complexity. For example, The complexity of latest available unscheduled channel with void filling (LAUC-VF) [7] is $O(m)$, where m is the number of voids. It is common that a system needs to keep track of 100 K to 1 million voids. Minimum starting void (Min-SV) [8], [9] takes $O(\log m)$ time to finish, which is a significant improvement over

the LAUC-VF. However, the Min-SV still requires $10 \log m$ memory accesses per burst scheduling request, which means that it can take up to a few microseconds to schedule a single burst.

B. Proportional QoS

The proportional QoS [6] keeps track of proportional burst loss probability among different priorities. A low-priority burst is intentionally dropped if the proportional differentiation rule is violated. Although this scheme is able to maintain proportional burst loss probabilities among different priorities, it encounters higher overall burst loss probability due to intentional dropping.

C. Early Drop and Wavelength Grouping

A burst early drop scheme based on the absolute QoS model was proposed in [10]. In burst early drop, an arriving burst will be dropped in a probabilistic manner if its predefined loss probability is violated, regardless of whether there is an idle channel. Similar to proportional QoS, this approach also causes excessive dropping, and the overall throughput is lower than a classless case.

In wavelength grouping [10], the wavelengths at the outgoing link were divided into wavelength groups. Each priority is limited to the use of no more than a maximum number of wavelengths. In static wavelength grouping, a fixed wavelength group is assigned to each priority class. In dynamic wavelength grouping, bursts are allowed to use wavelengths dynamically, as long as the total number of wavelengths occupied by a priority class does not exceed the predefined value.

By using wavelength grouping, it is easy to provide a guaranteed predefined target burst loss performance to each priority class. However, it is well known that the statistical multiplexing performance of large number of channels is much better than the performance of a collection of small number of channels. Therefore, the overall burst loss probability in wavelength grouping is higher than a system with complete wavelength sharing.

D. Look-Ahead Window (LAW)

The LAW [11], [12] resolves the burst contention by constructing a window of W time units. The bursts in the window are collectively considered in making scheduling decisions.

In LAW, the start time and the end time of the bursts are used to construct a directed graph. Service differentiation can be realized by assigning different weights to the edges in the directed graph. Bursts can be selected by finding the shortest path. Shortest path algorithms such as the Bellman–Ford algorithm can be used to solve the problem.

However, the complexity of the Bellman–Ford algorithm is $\Theta(|V| \bullet |E|)$, where $|V|$ is equal to twice the number of bursts in the LAW, and $|E|$ is equal to the number of bursts in the window. Therefore, the complexity of LAW is $\Theta(n^2)$, where n is the number of bursts in the LAW.

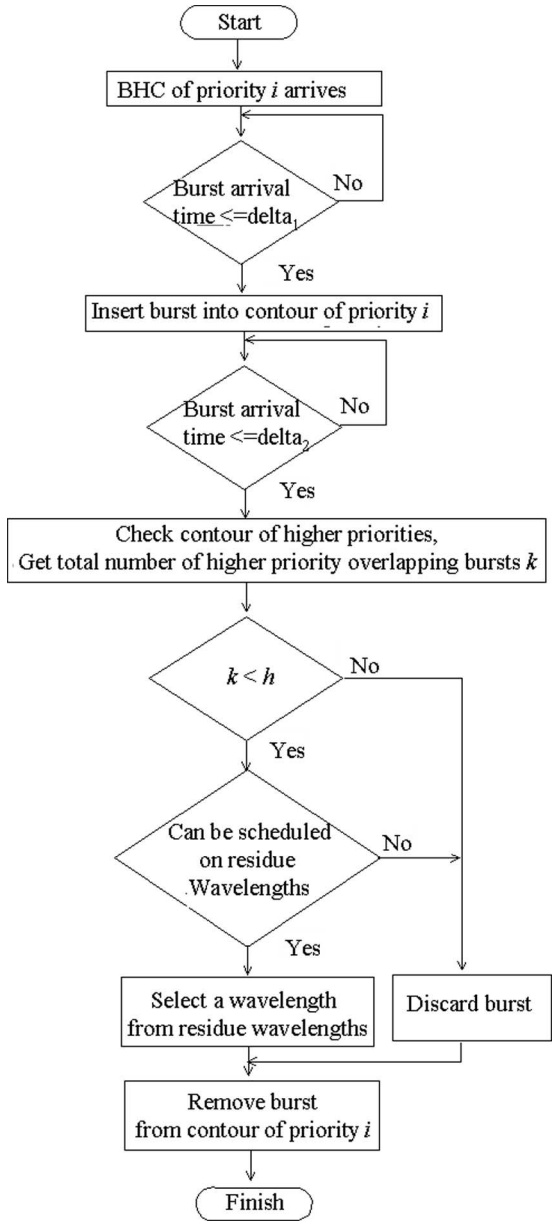

Fig. 4. Flow chart of the CBP algorithm.

IV. CBP Algorithm

In this paper, we propose a O(1) runtime CBP scheduling algorithm. The proposed algorithm provides the complete priority (class) isolation as defined in the offset-based QoS [5] where a high-priority burst is never blocked by a low-priority burst. In addition, a low-priority burst is only intentionally dropped by the scheduler if scheduling such a burst will cause an overlapping high-priority burst to be dropped. Therefore, there is no excessive dropping in CBP. More importantly, the CBP algorithm is the first hardware-based O(1) runtime priority algorithm proposed for OBS networks.

A. Algorithm Description

Assume that n priorities are supported. Assume that priority 0 has the highest priority. Each BHC carries a priority

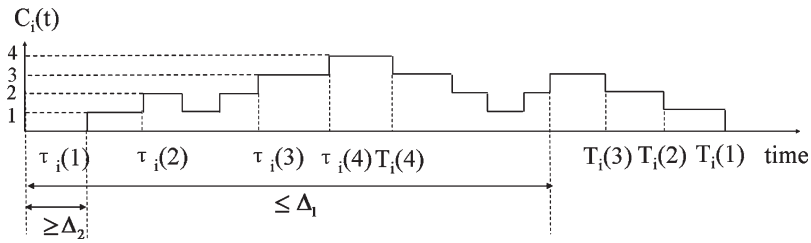

Fig. 5. Contour of priority i.

identifier, in addition to the normal BHC content fields such as the offset and the length. Assume that each link has h data channels. A contour is a projection of future burst occupancy.

Fig. 4 shows the flow chart of the proposed CBP algorithm. When a BHC of priority i arrives at an OBS router, the burst information carried in the BHC is recorded, and the BHC is immediately released to the next hop. The projected duration of the burst is added to the contour of priority i when the time difference between the current time and the projected burst arrival time is less than or equal to Δ_1 time units. The burst is scheduled by the scheduler when the time difference between the current time and the burst arrival time is less than Δ_2 time units. Once an outgoing wavelength channel is selected, such information is forwarded to the adjacent downstream router by a burst configuration packet. The release of BHC and the generation of the burst configuration packet are not shown in Fig. 4 since they are signaling protocol dependent and are not part of the core algorithm.

The bursts that have been added to the contour, but have not been scheduled, are referred to as pending bursts. Note that the delays of Δ_1 and Δ_2 time units are realized using electronic RAMs in the electronic control path of an OBS router. No FDL is needed in the optical data path in order to support the CBP algorithm.

We formally define the contour of priority i to be the function $C_i(t)$, which is equal to the number of pending priority i bursts which span time t. We say that a burst spans time t if it starts no later than t and ends after t. Based on the definition, we can plot the contour of each priority. Fig. 5 shows a typical contour of priority i, which plots the number of pending bursts of priority i at any time instance. A rising edge in the contour corresponds to the start of a burst. A falling edge in the contour corresponds to the end of a burst.

Let k denote the number of higher priority bursts that overlap with the burst to be scheduled. Define the dynamic set of residue wavelengths W_r as the wavelengths that would not be used by the higher priority overlapping bursts. Let r be the number of residue wavelengths

$$r = h - k, \qquad \text{if} \quad 0 \le k < h \tag{1.a}$$

$$r = 0, \qquad \text{if} \quad k \ge h. \tag{1.b}$$

Therefore, $W_r = \{w_0, w_1, \ldots, w_{r-1}\}$ if $r > 0$, and $W_r = \emptyset$ if $r = 0$. We also have $|W_r| = r$. If $r > 0$, the burst is allowed to choose from the set of residue wavelengths W_r. If the burst can find a wavelength from W_r that can accommodate the burst, the burst is scheduled. Otherwise, the burst is discarded. After the burst is scheduled or discarded, the duration of the burst is removed from the contour of its own priority.

There are two parameters in the algorithm, namely, Δ_1 and Δ_2. The parameter Δ_1 defines the latest time that a rising edge can occur in the contour, which corresponds to the latest start time of the last burst in the contour. The parameter Δ_2 defines the earliest time at which a rising edge can occur, which corresponds to the earliest start time of the first burst in the contour. The relationship of Δ_1 and Δ_2 with the contour is illustrated in Fig. 5. The difference in these two values defines the contour span. The additional discussion on the contour span can be found in Section IV-D.

From the high-level algorithm's perspective, no restriction needs to be placed on the values of Δ_1 and Δ_2, in which case, $\Delta_1 \to \infty$ and $\Delta_2 \to 0$. This means that a burst can be added to the contour when its BHC arrives at the router and can be scheduled when the burst is about to arrive. However, from the implementation's perspective, we need to place some constraints on Δ_1 and Δ_2 to ensure proper operations and efficient implementation. For example, in determining Δ_2, we need to take into account the burst scheduling time and processing queueing time. Proper value of Δ_1 allows the complex burst contour to be managed in O(1) time, as discussed in Section IV-C.

The modification to the signaling protocols and the control overhead associated with the CBP algorithm are explained as follows. In the CBP algorithm, the initial offset set by the ingress edge router has to be greater than Δ_1, which determines the desired contour span. This means that, instead of the commonly adopted hop-dependent offset which is equal to the per hop processing time multiplied by the number of hops, only a single network wide minimum offset needs to be maintained. Since, at each OBS router, the BHC is released immediately after the burst information is recorded, such an offset can be easily maintained throughout the OBS network without offset regeneration. The maintenance and updates on the contour do not impose any additional restrictions on the offset. Note that when the BHC is released to the next hop, an output channel has yet to be selected for the burst. Therefore, after the burst is scheduled, the selected wavelength needs to be communicated between adjacent routers in order for the routers to configure the optical interconnects properly. This additional signaling overhead is similar to the overhead in split processing [3] and is less than the overhead in dual-header OBS [13] since the configuration packets are only communicated between the adjacent OBS routers and are transparent to the ingress edge router.

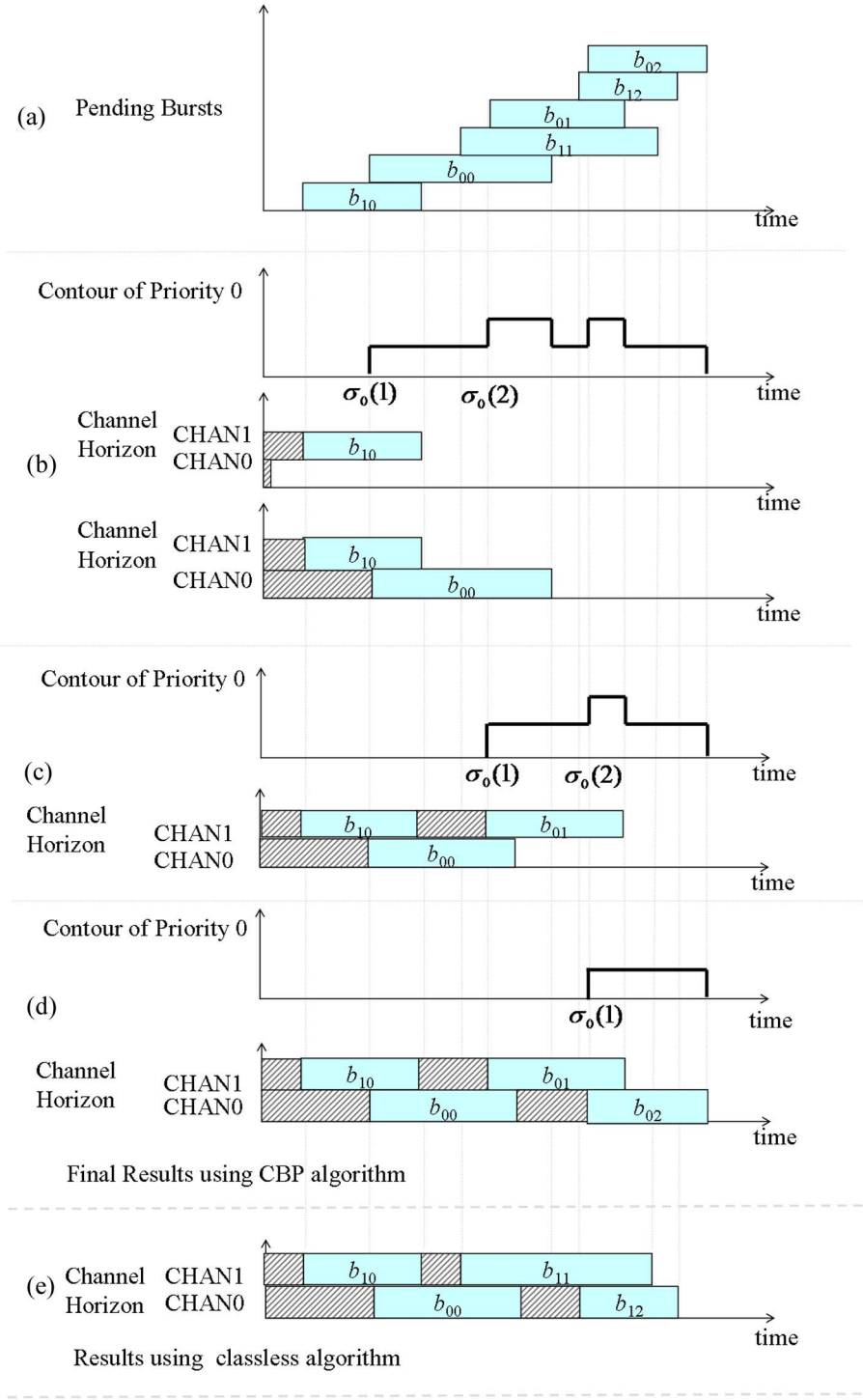

Fig. 6. Example of the CBP algorithm.

One possible drawback of the separation of the burst information recording phase (with the original BHC arrival) and the wavelength notification phase (with the burst configuration packet) is that downstream nodes may waste some bandwidth on the bursts that are discarded at upstream nodes. However, studies have shown that such effect on the overall performance is negligible [13]. One alternative signaling protocol is to hold the BHC until the burst is scheduled at Δ_2 time units ahead of the burst arrival time and release the BHC at that time. Although such a scheme will not need burst configuration packets between adjacent routers, the offset between the BHC and the burst has to be recreated using FDLs at the input of the OBS routers, which increases the system cost and the latency of the burst unnecessarily.

B. Examples

We use the following example to illustrate the operation of the CBP algorithm. The example shows a system with two priorities and two data channels per link. Bursts are numbered

as b_{ij}, where i is the priority identifier, and j is a sequence number for bursts in priority i. The sequence number is just for illustration purpose and is not part of the algorithm.

Fig. 6(a) shows all pending bursts. There are three bursts in each priority class that are waiting to be scheduled on the channel. The first plot in Fig. 6(b) shows the contour of priority 0. The contour of class 1 is not needed because class 1 is the lowest priority class. The following description focuses on the behavior of the scheduler.

The first burst to be scheduled is b_{10}. Since the burst belongs to the lower priority class (priority 1), the contour of priority 0 is examined before the burst is scheduled. In this case, b_{10} conflicts with one channel in the contour of priority 0. Therefore, the scheduler will leave one channel for the priority 0 burst yet to be scheduled and schedule b_{10} on the remaining residue wavelength. In this case, b_{10} can be admitted, and channel 0 is assigned to b_{10}.

At Δ_2 time units before the arrival time of b_{00}, b_{00} is scheduled. Since b_{00} belongs to the highest priority class, there is no need to check the contour. Channel 1 is assigned to b_{00}. After b_{00} is scheduled, the projected duration of b_{00} in the contour of priority 0 is removed. The updated contour is shown in the first plot in Fig. 6(c).

When it is time to schedule b_{11}, the duration of b_{11} is compared against the contour of priority 0. Since both channels will be occupied by priority 0, bursts for the duration of b_{11}, b_{11} are discarded.

When b_{01} is scheduled, the burst is directly scheduled onto channel 0 since it has the highest priority. The projection of b_{01} is removed from the contour of class 0.

When b_{12} is scheduled, its duration is compared with the contour shown in first chart in Fig. 6(d). Since it overlaps with one priority 0 burst in the contour, the scheduler tries to schedule b_{12} onto the residue wavelength. However, since the residue wavelength is busy when b_{12} arrives, b_{12} is discarded.

Finally, b_{02} is scheduled onto channel 0. The projected duration of b_{02} is removed from the contour of priority 0. The contour of priority 0 is empty from this point on.

Fig. 6(d) shows the final scheduling results using the CBP algorithm. As we can see, all three high-priority bursts are scheduled on channels successfully, despite the scheduling order of the bursts.

Fig. 6(e) shows the scheduling results of a classless algorithm. In this case, two high-priority bursts, namely, b_{01} and b_{02}, are discarded.

C. Analysis

In general, keeping an accurate contour requires complex data structures. In this paper, we propose a novel way of maintaining the priority contours in O(1) time.

A contour represents the number of channels that will be used by pending bursts of a particular priority. Because there can be, at most, h bursts being transmitted on an outgoing link at the same time, where h is the number of channels per link, there is no point in maintaining a contour with value greater than h. If adding a burst would result in $C_i(t) > h$, the burst is simply discarded. In this case, the burst is dropped because more than h bursts of the same priority want to use the wavelengths.

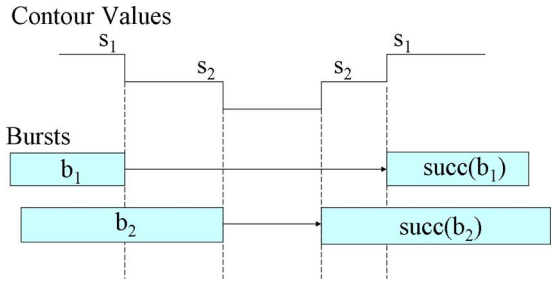

Fig. 7. Successor function.

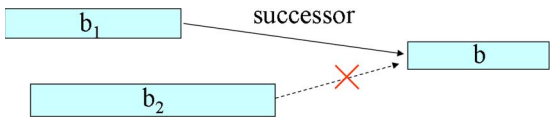

Fig. 8. Illustration of Theorem 1.

Assume that the start and finish times of the bursts are unique. In practice, if two values happen to be the same, we assume an infinitely small distance between the two values. Based on this assumption, there is a one-to-one correspondence between the start time of a burst and a rising edge in the contour and a one-to-one correspondence between the finish time of a burst and a falling edge in the contour.

The O(1) runtime operation of the CBP algorithm depends on some properties of the contours. To formalize these properties, let $R_i = \max_t C_i(t)$. Let $\tau_i(s)$ be the earliest time that $C_i(t) = s$, where $1 \le s \le R_i$. Let $T_i(l)$ be the latest time that $C_i(t) = l$, where $1 \le l \le R_i$. The corresponding points on a contour are illustrated in Fig. 5.

Property 1: $\tau_i(1) < \tau_i(2) < \cdots < \tau_i(R_i)$.
Property 2: $T_i(R_i) < T_i(R_i - 1) < \cdots < T_i(1)$.
Property 3: $C_i(\tau_i(j)) = j$, where $1 \le j \le R_i$.
Property 4: $C_i(T_i(j)) = j$, where $1 \le j \le R_i$.
Property 5: $T_i(R_i) > \tau_i(R_i)$.

These properties follow immediately after the definitions.

A burst is added to the contour of class i at Δ_1 time units before the burst arrival time. Let $b_i(t_1, t_2)$ be a burst to be added to the contour at time $t_1 - \Delta_1$, with a start time of t_1, a finishing time of t_2, and a priority of i.

A burst is scheduled on an outgoing channel by the scheduler at Δ_2 time units before the burst arrival time. Let $b'_i(t'_1, t'_2)$ be a burst to be scheduled by the scheduler at time $t'_1 - \Delta_2$, with a start time of t'_1, a finishing time of t'_2, and a priority of i.

Property 6: $t'_1 < \tau_j(k)$, where $1 \le k \le R_j$, for all $j \ne i$.

Proof: Time $\tau_j(k)$ corresponds to the start time of some pending bursts in the contour of priority other than i. Since the scheduler schedules the bursts in ascending order of the burst start times, t'_1 must be smaller than the start times of all bursts in the contour of all other classes. ■

Let b be a burst in the contour, and let s be the value of the contour function just before b ends. Define the successor function $\mathrm{succ}(b)$ to be the burst in the contour with the earliest start time for which the value of the contour function is equal to s, right after the burst starts. The successor function may

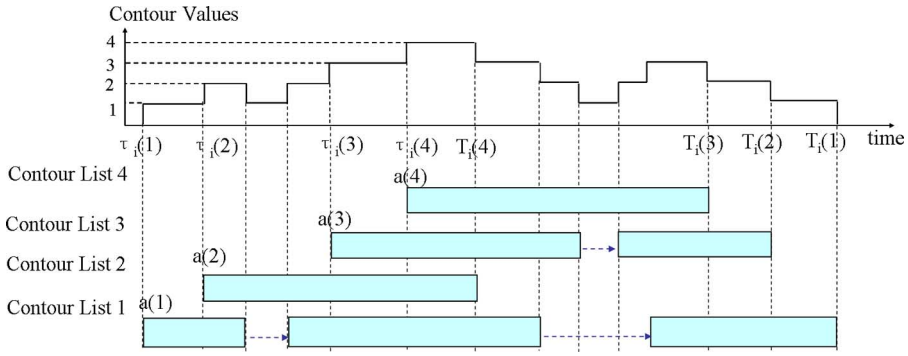

Fig. 9. Contour lists and their relationship with the contour.

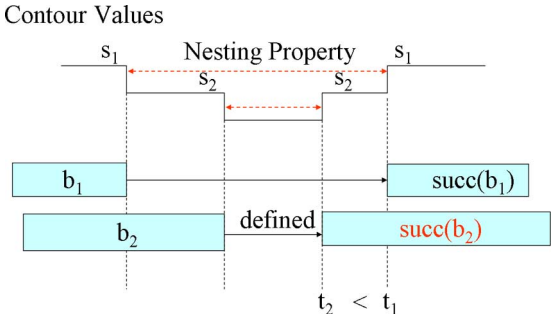

Fig. 10. Illustration of Theorem 2: the nesting property of the successor function.

be undefined for some bursts. Fig. 7 illustrates the successor function and its relationship with the contour.

Theorem 1: For any two bursts b_1 and b_2 in the contour, $\mathrm{succ}(b_1)$ is not equal to $\mathrm{succ}(b_2)$.

Proof: Let b_1 and b_2 be two bursts with $\mathrm{succ}(b_1) = \mathrm{succ}(b_2) = b$. The value of the contour just before the end of b_1 is equal to the value just before the end of b_2. Call this value s. Without loss of generality, assume that b_1 ends before b_2. Based on the definition of the successor function, b starts after b_2 ends. Because the value of the contour is s just before b_1 ends, the value of the contour is $s-1$ right after b_1 ends. Since the value of the contour is s again just before b_2 ends, there must be some burst that starts before b_2 ends that brings the value of the contour back to s. The existence of such a burst contradicts the assumption that $\mathrm{succ}(b_1) = b$. ■

Theorem 1 is illustrated in Fig. 8. Theorem 1 implies that the successor function partitions the bursts in the contour into a set of lists, which we call contour lists, as shown in Fig. 9.

Theorem 2 (Nesting Property of Successor Function): Let b_1 and b_2 be the bursts in two different lists defined by the successor function, and let b_1 end before b_2. If $\mathrm{succ}(b_1)$ is defined and starts after b_2 ends, then $\mathrm{succ}(b_2)$ is also defined, and $\mathrm{succ}(b_2)$ starts before $\mathrm{succ}(b_1)$ starts. Call this the nesting property of the successor function.

Proof: Let s_1 be the value of the contour just before b_1 ends, and let s_2 be the value of the contour just before b_2 ends. By the definition of the successor function, $C_i(t)$ is less than s_1 between the end of b_1 and the start of $\mathrm{succ}(b_1)$; therefore, $s_2 < s_1$. This means that there must be some time t after the end of b_2 and before the start of $\mathrm{succ}(b_1)$ when $C_i(t) = s_2$. The burst that starts at the earliest such time is the successor of b_2. ■

The nesting property of the contour function is illustrated in Fig. 10. Illustrations of Theorems 3 and 4 are shown in Fig. 11(a) and (b), respectively.

Theorem 3: If b is the first burst in a list defined by the successor function and starts at time t, then every list, which has a burst starting before t and a burst ending after t, has a burst that spans t.

Proof: Assume that there is a list L with a burst starting before t and a burst ending after t but no burst spanning t. Let b' be the latest burst in L that ends before t, and let b'' be the earliest burst in L that starts after t. As a result, there are no bursts in list L that start after b' ends and end before b'' starts. Burst b'' is the successor of b'. Since b starts before b'', $\mathrm{succ}(b') = b''$, and b' starts before b, by the nesting property of the successor function, successor function $\mathrm{succ}(*) = b$ must exist, which contradicts the fact that b is the first burst in its list. ■

Theorem 4: If b is the last burst in a list defined by the successor function and ends at time t, then every list, which has a burst starting before t and a burst ending after t, has a burst that spans t.

Proof: Assume that there is a list L with a burst starting before t and a burst ending after t but no burst spanning t. Let b' be the latest burst in L that ends before t, and let b'' be the earliest burst in L that starts after t. Therefore, there are no bursts in list L that start after b' ends and end before b'' starts. Burst b'' is the successor of b'. Since b ends after b' ends, $\mathrm{succ}(b') = b''$, and b'' starts after b ends, by the nesting property of the successor function, $\mathrm{succ}(b)$ must exist, which contradicts the fact that b is the last burst in its list. ■

Let b be a burst in the contour, and let k be the value of the contour function just after b starts. Based on the successor function, we can derive the predecessor function $\mathrm{pred}(b)$ to be the burst in the contour with the latest end time for which the value of the contour function is equal to k, right after the burst starts. The predecessor function may be undefined for some bursts.

Theorem 5 (Nesting Property of Predecessor Function): Let b_1 and b_2 be the bursts in two different lists defined by the successor function, and let b_1 start before b_2. If $\mathrm{pred}(b_2)$ is defined and ends before b_1 starts, then $\mathrm{pred}(b_1)$ is also defined, and $\mathrm{pred}(b_1)$ ends after $\mathrm{pred}(b_2)$ ends. Call this the nesting property of the predecessor function.

Proof: Let k_1 be the value of the contour just before b_1 starts, and let k_2 be the value of the contour just before b_2 starts. By the definition of the predecessor function, $C_i(t)$ is less than

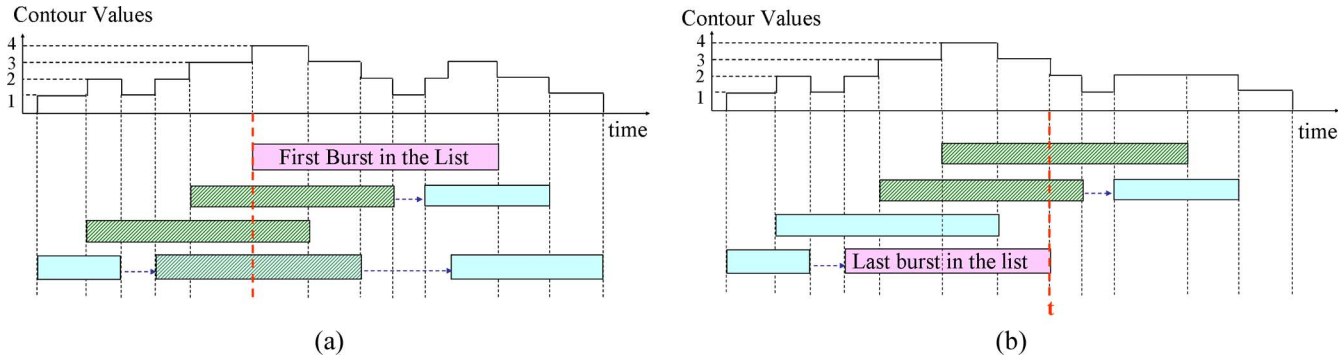

Fig. 11. (a) Illustration of Theorem 3. (b) Illustration of Theorem 4.

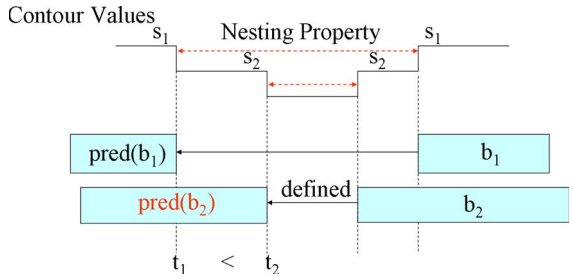

Fig. 12. Illustration of Theorem 5: the nesting property of the predecessor function.

k_2 between the end of $\mathrm{pred}(b_2)$ and the start of b_2; therefore, $k_2 > k_1$. This means that there must be some time t between the end of $\mathrm{pred}(b_2)$ and the start of b_1 when $C_i(t) = k_1$. The burst that starts at the earliest such time is the predecessor of b_1. ■

Fig. 12 shows the nesting property of the predecessor function. Fig. 13(a) and (b) illustrates Theorems 6 and 7, respectively.

Theorem 6: Let $a_i(k)$ be the start time of the first burst in contour list k, where the contour lists are ordered so that $a_i(1) < a_i(2) < \cdots$. For all contour lists, $C_i(t) < k$ for $t < a_i(k)$ and just before $a_i(k)$, $C_i(t) = k - 1$. Therefore, $\tau_i(k) = a_i(k)$, for $1 \le k \le R_i$, where R_i is the maximum contour value.

Proof: The property can be proved using induction.

For $k = 1$, it is obvious that before the start time of the first burst $a_i(1)$, the contour value is zero. Therefore, $C_i(t) < 1$ for $t < a_i(1)$. The value of the contour is one right after the burst starts. Therefore, $\tau_i(1) = a_i(1)$.

Assume that $C_i(t) < j$ for $t < a_i(j)$, and $\tau_i(j) = a_i(j)$ for $1 \le j < k$. We need to prove that $C_i(t) < k$ for $t < a_i(k)$; therefore, $\tau_i(k) = a_i(k)$.

Let b be the first burst in contour list k that starts at $a_i(k)$.

First, we prove that before b starts, the value of the contour is, at most, $k - 1$. Because bursts are added to the contour in their arrival order and b is the first burst in the kth list, there are only $k - 1$ lists before b is added to the contour. Since the bursts that are in the same list do not overlap in time based on the successor function, the value of the contour is, at most, $k - 1$ before b starts.

Next, we prove that the value of the contour is $k - 1$ just before b starts. To prove this, we first prove that all $k - 1$ lists end after b starts. This can be proved using contradiction.

Assume that some list ends before b starts. Let b' be the last burst in such a list with the latest end time that precedes the start time of b. Assume that the value of the contour is l right before b' ends. Right after b' ends, the contour value becomes $l - 1$. The start of b brings the value of the contour back to l. Based on the successor function, b is the successor of b', which contradicts the fact that b is the first burst in a list. Therefore, all $k - 1$ lists end after $a_i(k)$.

Because $a_i(1) < a_i(2) < \cdots < a_i(k-1) < a_i(k)$, all $k-1$ lists start before $a_i(k)$. Based on Theorem 3, each of the $k - 1$ lists has a burst that spans time $a_i(k)$. Therefore, the value of the contour is $k - 1$ just before b starts.

Therefore, $\tau_i(k) = a_i(k)$. ■

Theorem 7: Let $e_i(k)$ be the end time of the last burst in the contour list k where the contour lists are ordered so that $e_i(1) > e_i(2) > \cdots$. For all contours, $C_i(t) < k$ for $t > e_i(k)$ and immediately after $e_i(k)$, $C_i(t) = k - 1$. Therefore, $T_i(k) = e_i(k)$, for $1 \le k \le R_i$, where R_i is the maximum contour value.

Proof: This can be proved by induction.

For $k = 1$, we prove that $C_i(t) < 1$ for $t > e_i(1)$.

Assume that $C_i(t) \ge 1$ for some $t > e_i(1)$. There must exist some burst that ends after $e_i(1)$. Such a burst cannot be the last burst in the contour list since $e_i(1)$ corresponds to the latest ending time of the last burst in all contours. Therefore, the burst must be in the middle of some contour list. However, this cannot happen since the bursts in the contour lists are in time order based on the successor function. The ending time of a burst in the middle of the list must be earlier than the ending time of the last burst in the contour. Therefore, $C_i(t) < 1$ for $t > e_i(1)$.

Assume that $C_i(t) < j$ for $t > e_i(j)$, and $T_i(j) = e_i(j)$ for $1 \le j < k$. We need to prove that $C_i(t) < k$ for $t > e_i(k)$ and $C_i(t) = k - 1$ immediately after $e_i(k)$. Therefore, $T_i(k) = e_i(k)$.

Let b be the last burst in the contour that ends at $e_i(k)$.

First, we prove that the value of the contour is, at most, $k - 1$ after $e_i(k)$.

Since after time $e_i(k)$, there are only $k - 1$ contour lists. Because the bursts in the same contour list do not overlap in time based on the successor function, the value of the contour is, at most, $k - 1$.

Second, we prove that the value of the contour is $k - 1$ right after b ends. To prove this, we first prove that all $k - 1$ lists start before b ends. This can be proved using contradiction.

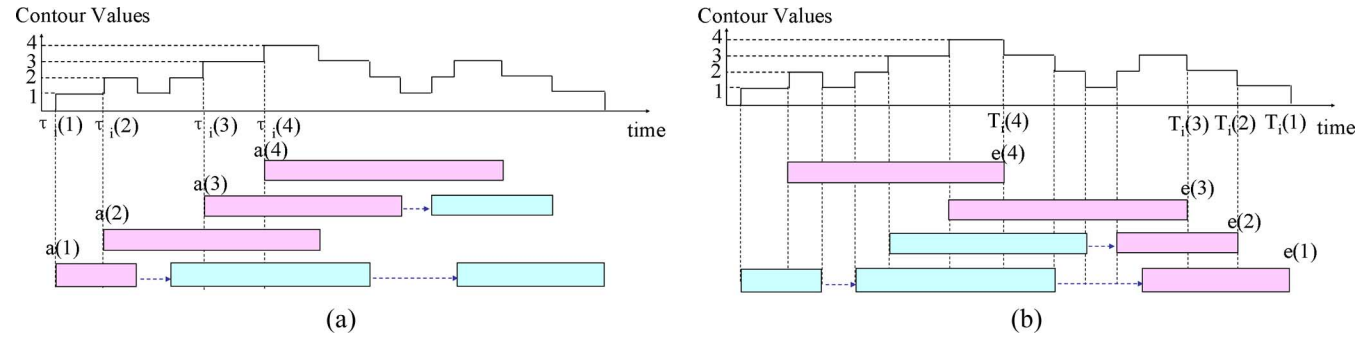

Fig. 13. (a) Illustration of Theorem 6. (b) Illustration of Theorem 7.

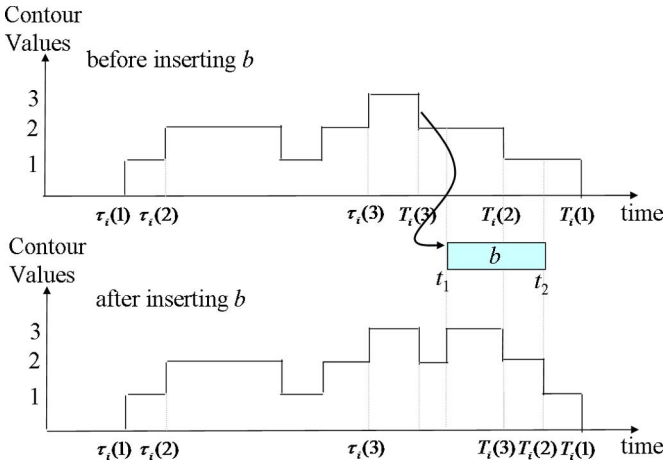

Fig. 14. Inserting a burst to the contour of priority i.

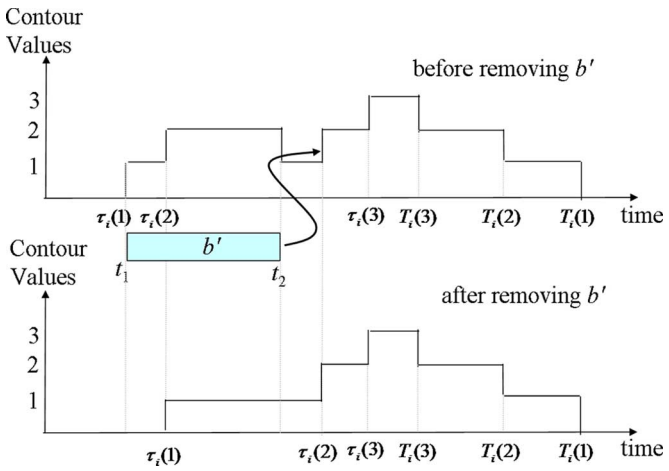

Fig. 15. Removing a burst from the contour of priority i.

Assume that there is some list that starts after b ends. Let b' be the first burst with the earliest start time in such a list. Assume that the value of the contour is l right before b ends. Therefore, right after b ends, the contour value is $l-1$. The start of b' brings the value of the contour back to l. Based on the successor function, b' is the successor of b, which contradicts the fact that b' is the first burst in a list. Therefore, all $k-1$ lists start before $e_i(k)$.

Because $e_i(1) > e_i(2) > \cdots e_i(k-1) > e_i(k)$, all $k-1$ lists end after $e_i(k)$. Based on Theorem 4, each of the $k-1$ lists has a burst that spans time $e_i(k)$. The value of the contour is $k-1$ immediately after $e_i(k)$.

Therefore, $T_i(k) = e_i(k)$. ■

Theorem 8: If the maximum value of the contour is R_i, there are, at most, R_i lists.

This immediately follows from Theorem 6.

Theorem 9: There are, at most, h lists, where h is the number of data channels per link.

Proof: Because the maximum value of the contour is h, there are, at most, h lists according to Theorem 8. ■

Based on the above theorems, inserting/removing a burst to/from the contour can be described in terms of contour list operations.

Fig. 14 shows an example of inserting a burst b that starts at t_1 and ends at t_2 in the contour of priority i. The value of the contour becomes three right after b starts. Based on the successor function, b becomes the successor of the burst whose ending time corresponds to the latest time that the contour value is three. Burst b goes to the end of that contour list. In the example, b becomes the successor of the burst whose ending time is $T_i(3)$.

More specifically, when a burst b is being inserted into the contour, we look at the last burst on each of the contour lists defined by the successor function. Let b_1 be the burst in this set that has the latest finishing time that precedes the start time of b. Then, $\mathrm{succ}(b_1) = b$, and b goes on the end of this list. If no such burst can be found, and the total number of nonempty lists is less than h, b starts a new list. If not, b is rejected.

After burst b is inserted, the contour values for the duration of b increase by one. The T_i values need to be updated accordingly. Specifically, all the T values that fall in the duration of b increase their indexes by one. The ending time of b replaces the latest T value that falls in the duration of b. In Fig. 14, $T_i(2)$ becomes the new $T_i(3)$, and the ending time t_2 becomes $T_i(2)$.

Fig. 15 illustrates an example of removing a burst b' that starts at t_1 and ends at t_2 from the contour of priority i. When burst b' is removed from the contour, the contour values decrease by one for the duration of b'. The value of the contour is two right before b' ends. Based on the successor function, the start time of the successor of b' is the earliest time that the contour is two again after b' ends. The start time of the successor of b' becomes the new $\tau_i(s)$. The indexes of the τ_i values which are smaller than the new $\tau_i(s)$ value decrease by one. In the case illustrated in Fig. 8, $\tau_i(2)$ becomes the new $\tau_i(1)$, and the start time of the successor of b' becomes the new $\tau_i(2)$.

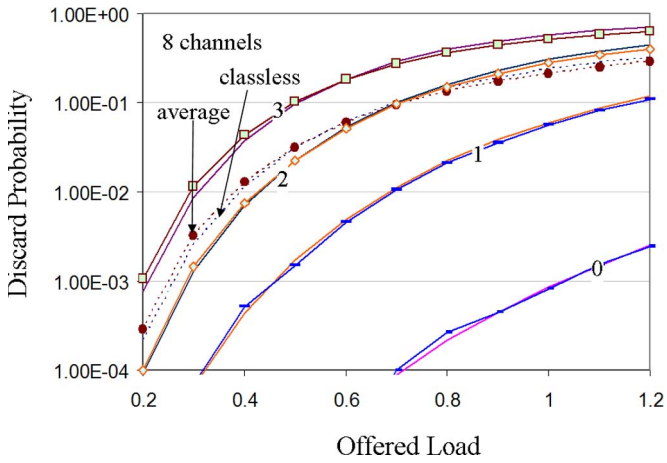

Fig. 16. Burst discard probability of the CBP algorithm.

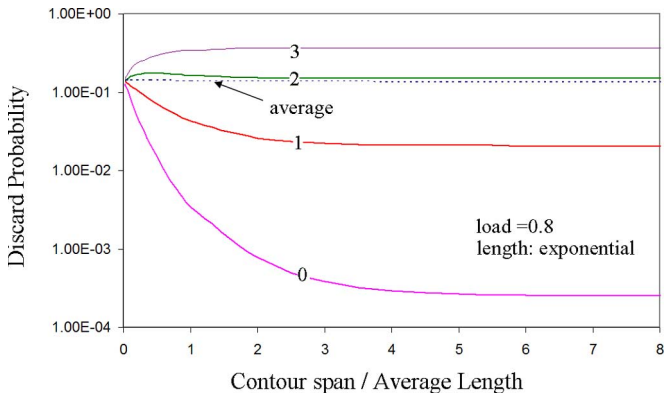

Fig. 17. Sensitivity to contour span (exponential).

As we can see above, we only need to access the first and last elements in the contour list. Therefore, the contour lists can be implemented using a set of FIFOs, which greatly reduced the complexity of maintaining an accurate contour. For each burst to be scheduled, there is a single FIFO write operation and a single FIFO read operation, which is an O(1) operation that involves two memory accesses.

D. Performance Evaluation

In addition to the analysis above, we have performed computer simulation on the performance of the CBP algorithm. The simulation results are compared to the analytical bound for complete priority isolation derived in [5]. The simulations are for four priority classes with equal load in each class. The average burst length is 100 μs with an exponential distribution. The offset has a fixed value of 600 μs. The value of Δ_1 is 500 μs, and the value of Δ_2 is 10 μs.

Fig. 16 shows the results from the simulation and the analytical bound. The curves with markers are the simulation results, and the curves without markers are from the analytic bound. The simulation results closely match the analytical bound. The average burst discard probability of the CBP algorithm is very close to that of a classless algorithm. The results have shown that the CBP algorithm can provide a complete priority isolation and does not incur unnecessary burst dropping.

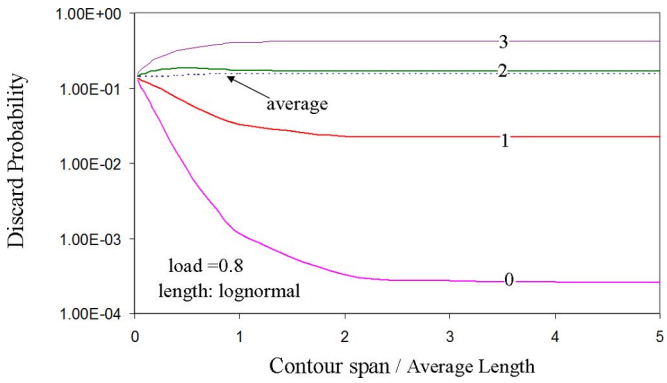

Fig. 18. Sensitivity to contour span (lognormal).

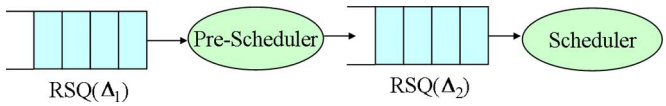

Fig. 19. CBP processing unit.

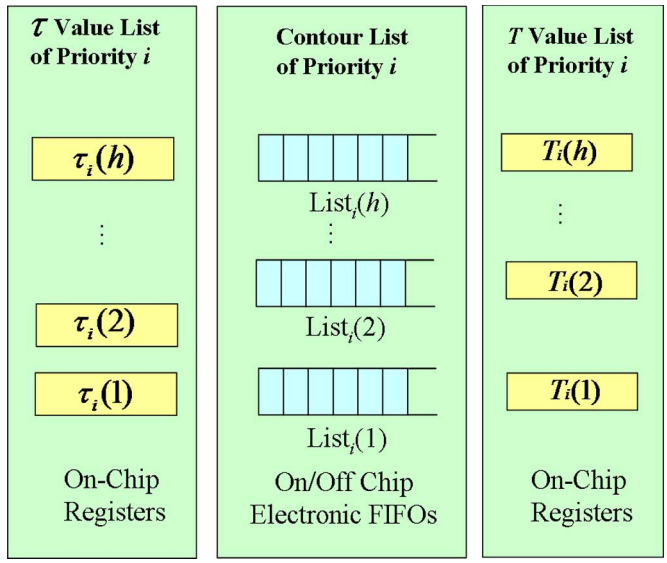

Fig. 20. Hardware structure for priority contour i.

The difference between Δ_1 and Δ_2 defines the contour span. If the length of a lower priority burst exceeds the contour span, it may detect higher priority bursts that overlap with the duration of the burst. The reason is that the higher priority burst will not have been added to the contour at the time a lower priority burst is scheduled. This results in priority inversion.

Fig. 17 shows the algorithm's sensitivity to the contour span when the burst length is exponentially distributed with an average of 100 μs. The x-axis is the ratio of the contour span to the average burst length. The value of Δ_2 is 10 μs. The discard probability flattens when the contour span is four times the average burst length. Therefore, if the contour span is four times the average burst length, we can ensure good priority isolation.

Fig. 18 shows the sensitivity to the contour span when the burst length has a lognormal distribution with an average of 100 μs and a standard deviation of 50 μs. The curves flatten at

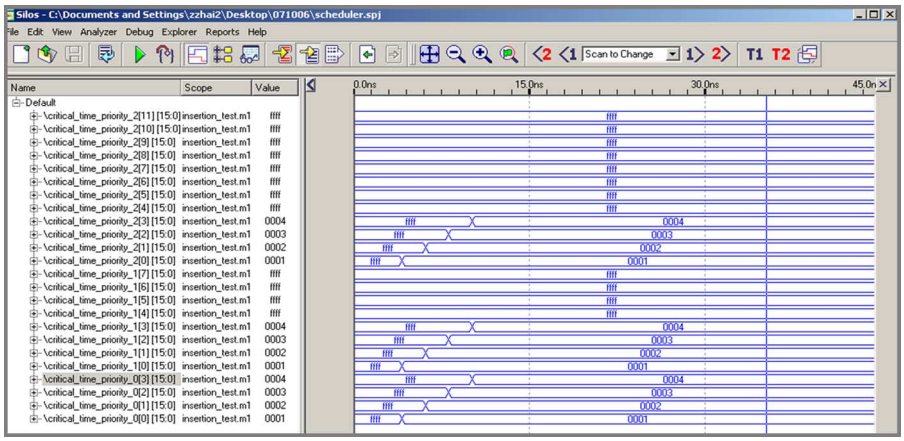

Fig. 21. Verilog HDL circuit simulation results.

twice the average burst length. It has been shown in [7] that the burst lengths for traffic aggregated at ingress routers approach a normal distribution. Therefore, a contour span of twice the average burst length is good enough to keep priority inversion to a minimum.

As mentioned in Section IV-A, Δ_1 will affect the network wide offset value. Therefore, proper value of Δ_1 needs to be selected with respect to the burst length to achieve a desired priority isolation.

E. Hardware Implementation

Define a resequencing buffer with parameter Δ to be a buffer that is capable of sorting the BHCs in ascending order of the burst arrival times and forwarding a BHC when the time difference between the current time and the burst arrival time is less than or equal to Δ time units. Such a resequencer is denoted as RSQ(Δ). An O(1) runtime resequencer design can be found in [14]. As mentioned previously, the resequencer is an electronic buffer implemented in the control path.

The hardware processing unit needed in implementing the CBP algorithm is illustrated in Fig. 19. The processing unit consists of a resequencing buffer with parameter Δ_1, a prescheduler, a resequencing buffer with parameter Δ_2, and a scheduler.

When a BHC arrives, the CBP processing unit copies of the burst information carried in the BHC and releases the original BHC immediately to the next hop. For the rest of this paper, we use BHC and the burst information in the BHC interchangeably. The BHC is placed in the first resequencing buffer where it stays until the time difference between the current time and the projected burst arrival time is less than or equal to Δ_1 time units. The BHC is then processed by the prescheduler. The prescheduler inserts the projected burst duration in the contour of the priority to which the burst belongs, using the offset and length fields in the BHC, and places the BHC in the second resequencer.

The BHC stays in the second resequencer until Δ_2 time units before the burst arrival time. The BHC is then processed by the scheduler. The scheduler makes scheduling decisions based on the status of the contours and outgoing channels. A basic horizon scheduler [3] can be used to keep track of the channel usage. After a BHC is scheduled, the projection of the burst is removed from the contour of its priority.

The CBP algorithm is suitable for cost-effective high-speed hardware implementation. Fig. 20 shows the hardware structure for priority contour i. The structure consists of h contour lists, which are implemented as FIFOs. To facilitate high-speed processing, the first and last elements in each FIFO are stored in on-chip registers. The FIFOs can be stored in off-chip memories.

There are three types of operations performed on the contour, namely, inserting a burst into the contour, removing a burst from the contour, and burst scheduling. Apparently, inserting and removing an element to/from an FIFO is an O(1) operation.

Note that during the scheduling phase, the scheduler only needs to access the first element in each contour list, which is stored in on-chip hardware registers and is readily available. Therefore, the complexity of scheduling a burst is independent of the number of pending bursts in the contour.

The comparison in determining the number of residue wavelengths can be done in constant time (a few clock cycles) using hardware for any practical values of channels and priorities. Once the number of residue wavelengths is determined, the scheduling of the burst can be done in O(1) time in hardware [15]. In addition, updating the T_i values and τ_i values in the contour only involves a single clock cycle shift register operation.

As a result, the proposed CBP algorithm can achieve O(1) runtime in hardware. The proposed CBP algorithm has been implemented in hardware using Verilog HDL (Hardware Description Language). The circuit simulation results that show O(1) operations are shown in Fig. 21.

V. Conclusion

In this paper, we have proposed a novel O(1) runtime CBP algorithm. The algorithm provides complete priority isolation using priority contours without encountering extra offset time or excessive burst dropping. The proposed CBP algorithm is the first O(1) runtime priority algorithm that is well suited to high-speed hardware implementation.

REFERENCES

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

[2] 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, Mar. 1999.

[3] Y. Chen and J. S. Turner, "WDM burst switching for petabit capacity routers," in *Proc. IEEE Military Commun. Conf.*, 1999, pp. 968–973.

[4] J. Y. Wei and R. McFarland, "Just-in-time signalling for WDM optical burst switching networks," *J. Lightw. Technol.*, vol. 18, no. 12, pp. 2019–2037, Dec. 2000.

[5] M. Yoo and C. Qiao, "QoS performance of optical burst switching in IP-over-WDM networks," *IEEE J. Sel. Areas Commun.*, vol. 18, no. 10, pp. 2062–2071, Oct. 2000.

[6] Y. Chen, M. Hamdi, and D. H. K. Tsang, "Proportional QoS over OBS networks," in *Proc. Globecom*, 2001, vol. 3, pp. 1510–1514.

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

[8] J. Xu, C. Qiao, J. Li, and G. Xu, "Efficient channel scheduling in optical burst switched networks," in *Proc. IEEE INFOCOM*, 2003, vol. 3, pp. 2268–2278.

[9] J. Xu, C. Qiao, J. Li, and G. Xu, "Efficient burst scheduling algorithms in optical burst-switched networks using geometric techniques," *IEEE J. Sel. Areas Commun.*, vol. 22, no. 9, pp. 1796–1881, Nov. 2004.

[10] Q. Zhang, V. Vokkarane, B. Chen, and J. P. Jue, "Early drop and wavelength grouping schemes for providing absolute QoS differentiation in optical burst-switched networks," in *Proc. Globecom*, Dec. 2003, vol. 5, pp. 2694–2698.

[11] F. Farahmand and J. P. Jue, "Look-ahead window contention resolution in optical burst switched networks," in *Proc. Workshop HPSR*, Jun. 2003, pp. 147–151.

[12] F. Farahmand and J. P. Jue, "Supporting QoS with look-ahead window contention resolution in optical burst-switched networks," in *Proc. IEEE Globecom*, San Francisco, CA, Dec. 2003, pp. 2699–2703.

[13] N. Barakat and E. H. Sargent, "Dual-header optical burst switching, a new architecture for WDM burst-switched networks," in *Proc. INFOCOM*, 2005, pp. 685–693.

[14] Y. Chen, J. S. Turner, and P.-F. Mo, "Optimal wavelength scheduling in optical burst-switched networks," *J. Lightw. Technol.*, vol. 25, no. 8, Aug. 2007.

[15] Y. Chen, J. S. Turner, and Z. Zhai, "Design and implementation of an ultra fast pipelined wavelength scheduler for optical burst switching," in *Proc. IASTED Int. Conf. OCSN*, Jul. 2006, pp. 263–270.

Yuhua Chen (M'04) received the B.S. degree in electronic engineering from Fudan University, Shanghai, China, in 1995 and the M.S. degree in computer science, the M.S. degree in electrical engineering, and the D.Sc. degree in electrical engineering from Washington University in St. Louis, St. Louis, MO, in 1997, 1998, and 2003, respectively.

She was a Research Associate/Hardware Design Engineer with the Applied Research Laboratory (ARL), Washington University in St. Louis, from 1999 to 2004, working on advanced networking system design and prototyping. She was one of the original developers for optical burst switching. She is currently an Assistant Professor with the Department of Electrical and Computer Engineering, University of Houston, Houston, TX. Her research interests include optical burst switching, high-performance routers, system prototyping, reconfigurable systems, system-on-chip, and wireless sensor networks.

Jonathan S. Turner (M'77–SM'88–F'90) received the M.S. and Ph.D. degrees in computer science from the Northwestern University, Evanston, IL, in 1979 and 1981, respectively.

He holds the Henry Edwin Sever Chair of Engineering at Washington University in St. Louis, St. Louis, MO, where he is the Director of the Applied Research Laboratory. His primary research interest is the design and analysis of switching systems, with special interest in systems supporting multicast communication. He has been awarded more than 20 patents for his work on switching systems and has many widely cited publications.

Dr. Turner is a member of the Association for Computing Machinery (ACM) and the Society of Fire Protection Engineers (SIAM). He received the Koji Kobayashi Computers and Communications Award from the IEEE in 1994 and the IEEE Millennium Medal in 2000.

Zhi Zhai (S'06) received the B.S. degree in electrical engineering from Nankai University, Tianjin, China, in 2005. He is currently working toward the Ph.D. degree with the Department of Electrical and Computer Engineering, University of Houston, Houston, TX.

His research interests include optical networks and wireless sensor networks.