# A Simple Proof for the Constructions of Optical Priority Queues

Hsien-Chen Chiu, Cheng-Shang Chang, Jay Cheng, and Duan-Shin Lee
Institute of Communications Engineering
National Tsing Hua University
Hsinchu 300, Taiwan, R.O.C.
Email: hcchiu@gibbs.ee.nthu.edu.tw
cschang@ee.nthu.edu.tw
jcheng@ee.nthu.edu.tw
lds@cs.nthu.edu.tw

**Abstract**

Constructions of optical queues by optical Switches and fiber Delay Lines (SDL) have received a lot of attention lately. In this short paper, we provide a simple proof for the construction of a priority queue with a switch and fiber delay lines in Sarwate and Anantharam [10]. Our proof not only gives the insights needed to understand why such a construction works, but also leads to a much general result that recovers the result in [10] as a special case. Moreover, our result can be easily extended to the constructions of priority queues with multiple inputs. Such constructions appear to be new in the literature.

**Index Terms**

Priority queues, optical switches, switched delay lines.

This research is supported in part by the National Science Council, Taiwan, R.O.C., under Contract NSC-93-2213-E-007-040, Contract NSC-93-2213-E-007-095, Contract NSC-94-2213-E-007-046, and the Program for Promoting Academic Excellence of Universities NSC 94-2752-E-007-002-PAE.

## I. Introduction

Queues are commonly used as a mean to resolve conflicts for people who compete for the same resources. Constructing a queue for people might be as easy as writing down a name list. However, constructing a queue for non-stopping photons appears to be a challenging problem. The need for constructing optical queues that store optical packets (composed of a train of photons) is driven by the recent advances in optical transmission technologies. As the transmission speed of optical links has been increased so rapidly, the traditional approach of converting optical packets back to electronic packets and storing packets in electronic memories becomes very costly.

One common approach to construct optical queues is to use optical Switches and fiber Delay Lines (SDL). The idea of using SDL for constructing an optical queue is to route (non-stopping) optical packets through a series of fiber delay lines so that optical packets depart from the queue at the *right* time. Early works, mainly focusing on the feasibility of SDL, include the feedback system by Karol [1] and the CORD (COntention Resolution by Delay lines) project [2]. In the past decade, researchers have made significant progresses in the mathematical theory of constructing several types of optical queues, including multiplexers in [3], [4], [5], [6], [7], FIFO queues in [8], non-overtaking delay lines and flexible delay lines in [9], and priority queues in [10].

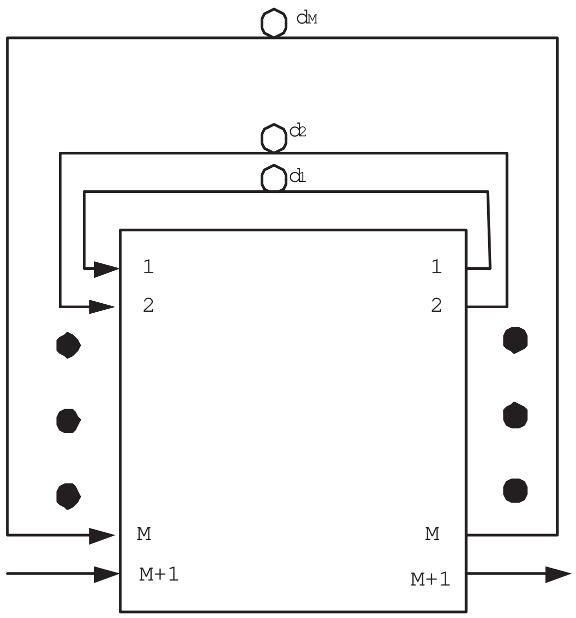


Fig. 1. A construction of a priority queue via a single switch and fiber delay lines.

Our main interest of this paper is the constructions of optical priority queues. Sarwate and Anantharam [10] considered a feedback system in [1] (see Figure 1). In such a feedback system, there is an $(M+1)\times(M+1)$ crossbar switch and $M$ fiber delay lines with delays $d_i$, $i=1,2,\ldots,M$. If $M=2k-1$ for some positive integer $k$, $d_i = i$ for $i = 1, \dots, k$, and $d_i = 1$ for $i = k+1, \dots, 2k-1$, then it was shown in [10] that such a system can be used for exact emulation of a priority queue with buffer $\sum_{i=1}^{k} d_i$. Our main contribution in this paper is to provide a much simpler and shorter proof than that given in [10]. Our proof not only gives the insights needed to understand why such a construction works, but also leads to a much general result that recovers the choice of the delays of the $M$ fiber delay lines in [10] as a special case. Moreover, our result can be easily extended to the constructions of priority queues with *multiple inputs*. Such constructions appear to be new in the literature.

## II. Priority Queues and Complementary Priority Queues

As our constructions of optical queues are based on optical Switches and fiber Delay Lines (SDL), we first introduce some basic assumptions and concepts for SDL. As in most papers on SDL in the literature, we consider fixed size packets over optical links. Assume that time in all our optical links is slotted and synchronized so that a packet can be transmitted within a time slot. A fiber delay line with delay $d$ is an optical link that requires $d$ time slots for a packet to traverse through. Specifically, if a packet arrives at the input of a fiber delay line with delay $d$ at time $t$ (the $t^{th}$ time slot), the packet will depart from the fiber delay line at time $t + d$. An $M \times M$ crossbar switch is a network element with $M$ input links and $M$ output links that realizes all the $M!$ permutations between its inputs and outputs.

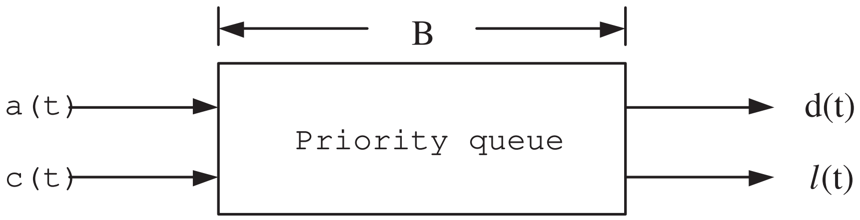


Fig. 2. A priority queue with buffer $B$.

In the following, we define a (discrete-time) priority queue with buffer $B$.

**Definition 1 (Priority queue)** *A priority queue with buffer $B$ is a network element that has one input link, one control input link, and two output links (see Figure 2). One output link is for departing packets and the other is for lost packets. When a packet arrives at the queue, it is associated with a label, called* priority. *We assume that there is a total order for the priorities of all the packets. As shown in Figure 2, let $c(t)$ be the state of the control input at time t. When $c(t) = 1$, we say the priority queue is enabled at time t. On the other hand, the priority queue is disabled at time t if $c(t) = 0$. Also, let $a(t)$ be the set of the packet arriving at time t (if any[1]), $d(t)$ be the set of the packet departing at time t (if any), $\ell(t)$ be the set of the lost packet*

[1] This means that $a(t)$ is an empty set if there is no packet arriving at time $t$, and is a singleton otherwise.

*at time $t$ (if any), and $q(t)$ be the set of packets queued at the priority queue at time $t$ (at the end of the $t^{th}$ time slot). Then the priority queue with buffer $B$ satisfies the following five properties:*

(P1) *Flow conservation: arriving packets from the input link are either stored in the buffer or transmitted through the two output links, i.e.,*

$$q(t) = q(t-1) \cup a(t) \backslash (d(t) \cup \ell(t)). \tag{1}$$

(P2) *Non-idling: if the control input is enabled, i.e., $c(t) = 1$, then there is always a departing packet if there are packets in the buffer or there is an arriving packet, i.e.,*

$$|d(t)| = \begin{cases} 1 & \text{if } c(t) = 1 \text{ and } |q(t-1) \cup a(t)| > 0 \\ 0 & \text{otherwise} \end{cases} . \tag{2}$$

(P3) *Maximum buffer usage: if the control input is not enabled, i.e., $c(t) = 0$, then there is a lost packet only when buffer is full and there is an arriving packet, i.e.,*

$$|\ell(t)| = \begin{cases} 1 & \text{if } c(t) = 0 \text{ and } |q(t-1) \cup a(t)| > B \\ 0 & \text{otherwise} \end{cases} . \tag{3}$$

(P4) *Priority departure: if there is a departing packet at time $t$, the departing packet is the one with the highest priority among all the packets in $q(t-1) \cup a(t)$.*

(P5) *Priority loss: if there is a lost packet at time $t$, the lost packet is the one with the lowest priority among all the packets in $q(t-1) \cup a(t)$.*

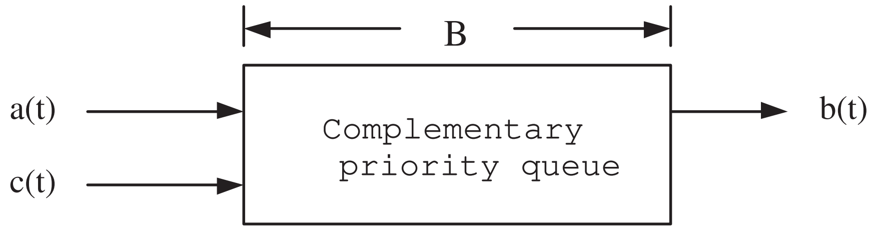


Fig. 3. A complementary priority queue.

To construct a priority queue, one needs to verify the five properties (P1)–(P5) in Definition 1. In the following, we introduce a complementary priority queue that reduces these five properties into two simple properties. As such, it is much easier to verify a construction of a complementary priority queue.

**Definition 2 (Complementary priority queue)** *A complementary priority queue with buffer $B$ is a network element that has one input link, one control input link, and one output link (see Figure 3). As in a priority queue, every packet is associated with a label, called* priority, *and there is a total order for the priorities. At time 0, there are $B$ packets stored in the network element. Unlike a priority queue, there is always an* *arriving packet and a departing packet in every time slot. As shown in Figure 3, let $c(t)$ be the state of the control input, $a(t)$ be the set of the packet arriving at time $t$, $b(t)$ be the set of the packet departing at time $t$, and $q^c(t)$ be the set of packets queued at the complementary priority queue at time $t$ (at the end of the $t^{th}$ time slot). Then the complementary priority queue with buffer $B$ satisfies the following two properties:*

(C1) *Flow conservation: arriving packets from the input link are either stored in the buffer or transmitted through the the output link, i.e.,*

$$q^c(t) = q^c(t-1) \cup a(t) \backslash b(t). \tag{4}$$

(C2) *Complementary priority departure: if $c(t) = 1$, then the departing packet is the one with the* highest *priority among all the packets in $q^c(t-1) \cup a(t)$. On the other hand, if $c(t) = 0$, then the departing packet is the one with the* lowest *priority among all the packets in $q^c(t-1) \cup a(t)$.*

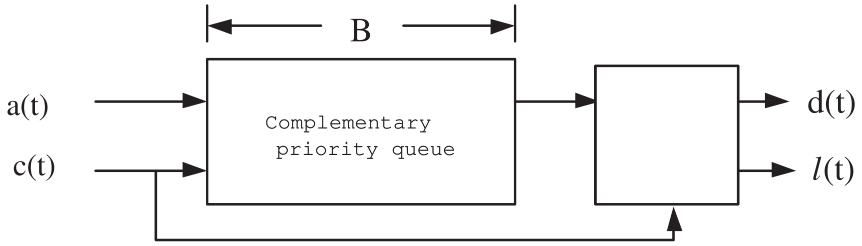


Fig. 4. A construction of a priority queue with buffer $B$ via a concatenation of a complementary priority queue with buffer $B$ and a $1 \times 2$ switch.

Clearly, a complementary priority queue and a priority queue are closely related. This is further clarified in Proposition 3 below.

**Proposition 3** *As shown in Figure 4, a priority queue with buffer $B$ can be constructed by a concatenation of a complementary priority queue with buffer $B$ and a $1 \times 2$ switch.*

**Proof.** The key is to view empty time slots as *fictitious* packets that have priorities lower than those of real packets. Moreover, the priorities among the fictitious packets are decreasing in the order of their arrival times. As such, we have a total order among all the packets, including both the real packets and the fictitious packets. To emulate an empty priority queue at time 0, we can store $B$ fictitious packets in the complementary priority queue.

Now we consider the following two cases.

Case 1. $c(t) = 1$:

In this case, we connect the input of the $1 \times 2$ switch to $d(t)$ in Figure 4. The complementary priority queue selects the packet with the highest priority among all the packets in $q^c(t-1) \cup a(t)$, where $q^c(t-1)$ is the set of packets stored in the complementary priority queue at the time $t-1$. If there is a real packet in

$q^c(t-1) \cup a(t)$, then $d(t)$ contains a real packet as fictitious packets have priorities lower than those of real packets. Moreover, this packet is the one with the highest prioity. Thus, (P2) and (P4) in Definition 1 are satisfied.

Case 2. $c(t) = 0$:

In this case, we connect the input of the $1 \times 2$ switch to $\ell(t)$ in Figure 4. The complementary priority queue selects the one with the lowest priority among all the packets in $q^c(t-1) \cup a(t)$. If there is a fictitious packet in $q^c(t-1) \cup a(t)$, then $\ell(t)$ contains a fictitious packet (an empty time slot) as fictitious packets have priorities lower than those of real packets. If there is no fictitious packet in $q^c(t-1) \cup a(t)$, then $\ell(t)$ contains a real packet and this real packet is the one with the lowest priority. Thus, (P3) and (P5) in Definition 1 are also satisfied. ■

## III. A Construction of a Complementary Priority Queue

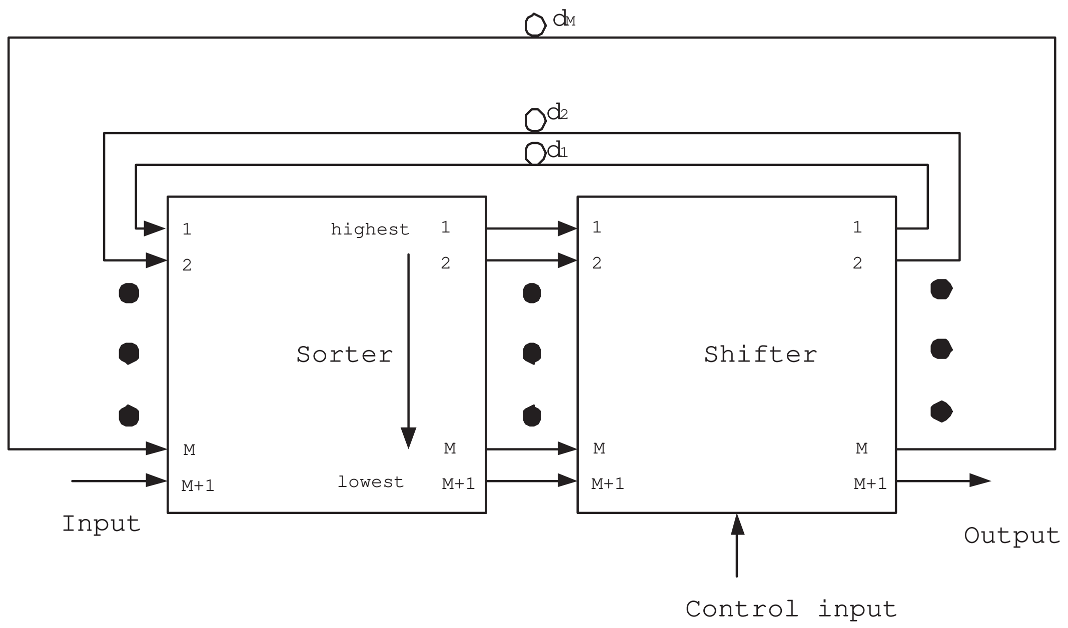


Fig. 5. A construction of a complementary priority queue with buffer $\sum_{i=1}^{M} d_i$.

In Figure 5, we show a construction of a complementary priority queue with buffer $\sum_{i=1}^{M} d_i$. In our construction, there are two $(M+1) \times (M+1)$ crossbar switches: a sorter (on the left hand side) and a shifter (on the right hand side). The key insight of our construction is based on the following assumption:

(A1) All the packets stored in all the fiber delay lines in Figure 5 cannot be either the packet with the highest priority or the packet with the lowest priority until they appear at the inputs of the sorter.

If (A1) holds, then the packets that appear at the inputs of the sorter contain both the packet with the highest priority and the packet with the lowest priority. The sorter then sorts the packets at the $M+1$ inputs in the order of their priorities. By so doing, the first output from the sorter is the packet with the highest priority and the $(M+1)^{th}$ output from the sorter is the packet with the lowest priority.

The $(M+1) \times (M+1)$ switch on the right hand side is a shifter that only has two connection patterns. When $c(t) = 0$, its connection pattern is realized by the $(M+1) \times (M+1)$ identity matrix, i.e., the matrix $I = (I_{ij})$ with $I_{i,j} = 1$ for $i = j$ and $I_{i,j} = 0$ for $i \neq j$. As such, the $(M+1)^{th}$ input of the shifter is connected to the $(M+1)^{th}$ output of the shifter and the packet with the lowest priority is sent out from the output link. On the other hand, when $c(t) = 1$, its connection pattern is realized by the $(M+1) \times (M+1)$ circular-shift matrix, i.e., the matrix $P = (P_{ij})$ with $P_{i,j} = 1$ for $i = (j \bmod (M+1)) + 1$ and $P_{i,j} = 0$ otherwise. As such, the first input of the shifter is connected to the $(M+1)^{th}$ output of the shifter and the packet with the highest priority is sent out from the output link. Thus, the construction emulates a complementary priority queue if (A1) holds.

The $M$ outputs of the shifter, indexed from $i = 1, \ldots, M$, are connected back to the the corresponding $M$ inputs of the sorter via $M$ fiber delay lines with delays $d_i$, $i = 1, \ldots, M$. For a fiber delay line with delay $d$, there are $d$ packets stored in that delay line. As such, there are $\sum_{i=1}^{M} d_i$ packets stored in the $M$ fiber delay lines. The question is then how we choose $d_i$'s so that the assumption in (A1) holds. This is answered in Proposition 4 below.

**Proposition 4** *Suppose that (A1) holds at time 0. If* $0 < d_i \leq \min[i, M+1-i]$ *for all* $i = 1, 2, \ldots, M$, *then the construction in Figure 5 is a complementary priority queue with buffer* $\sum_{i=1}^{M} d_i$.

**Proof.** It suffices to argue by induction that (A1) holds for all time. Suppose that (A1) holds up to time $t-1$. As such, we have exactly emulated a complementary priority queue with buffer $\sum_{i=1}^{M} d_i$ up to time $t$. For both cases $c(t) = 0$ and $c(t) = 1$, we also know that the priorities of the packets at the $M$ outputs of the shifter, indexed from $1, 2, \ldots, M$, are decreasing. Let us consider the packet at the $i^{th}$ output of the shifter. Call this packet the tagged packet. For the tagged packet, there are $i-1$ packets that have priority higher than its priority and there are $M-i$ packets that have priority lower than its priority. As the construction departs a packet in a time slot, the tagged packet cannot be the packet with the highest priority or the packet with the lowest priority for the next $\min[i-1, M-i]$ time slots. As the delay of the $i^{th}$ delay line $d_i$ is less than or equal to $\min[i, M+1-i]$, it follows that the tagged packet cannot be either the packet with the highest priority or the packet with the lowest priority until it appears at the input of the sorter. Thus, the assumption (A1) holds at time $t$. ■

We first note that the purpose of having two $(M+1) \times (M+1)$ crossbar switches in our construction is for the ease of the presentation and the proof. In practice, one can combine these two switches into one to reduce the hardware cost. Also, note that for $M = 2k-1$, the maximum buffer size that can be achieved by Proposition 4 is to set $d_i = i$ for $i = 1, 2, \ldots, k$, and $d_i = 2k - i$ for $i = k+1, k+2, \ldots, 2k-1$. For this, we have buffer size $\sum_{i=1}^{2k-1} d_i = k^2$. And for $M = 2k$, the maximum buffer size that can be achieved by Proposition 4 is to set $d_i = i$ for $i = 1, 2, \ldots, k$, and $d_i = 2k+1-i$ for $i = k+1, k+2, \ldots, 2k$. For this, we have buffer size $\sum_{i=1}^{2k} d_i = k^2 + k$. One less efficient way, as originally proposed in [10], is to choose $M = 2k-1$ and set $d_i = i$ for $i = 1, 2, \ldots, k$, and $d_i = 1$ for $i = k+1, k+2, \ldots, 2k-1$, which gives buffer size $\sum_{i=1}^{k} d_i = (k^2+k)/2$. As shown in Proposition 3, a complementary priority queue (in conjunction with a $1 \times 2$ switch) can be used for emulating a priority queue. Proposition 4 recovers the result in [10] as a special case. Finally, we note that if one would like to drop the arriving packet when the buffer is full, one can simply add a $1 \times 2$ switch at the input as discussed for FIFO queues in [8].

## IV. Extensions to Priority Queues with Multiple Inputs

In this section, we extend our result to priority queues with multiple inputs. For this, let us consider an $N$-to-1 priority queue with buffer $B$. For such a queue, there are $N$ input links, one control input $c(t)$, and $N+1$ output links. The $N+1$ output links consist of one output link for departing packets and $N$ output links for lost packets.

The definition of an $N$-to-1 priority queue is basically the same as that in Definition 1. An $N$-to-1 priority queue also satisfies the flow conservation property in (P1), the non-idling property in (P2) and the priority departure property in (P4). The key difference is that there might be multiple packet losses in a time slot as there are multiple packet arrivals in a time slot. For this, we need to modify the maximum buffer usage property in (P3) and the priority loss property in (P5). Let $c(t)$ be the state of the control input, $q(t)$ be the set of packets stored in the buffer at time $t$ and $a(t)$ be the set of packets arriving at time $t$. If $|q(t-1) \cup a(t)| - c(t) > B$, then there are $|q(t-1) \cup a(t)| - c(t) - B$ lost packets at time $t$. When there are $\ell$ lost packets in a time slot (for some $\ell \geq 1$), these $\ell$ lost packets are selected from the $\ell$ lowest priority packets among the packets in $q(t-1) \cup a(t)$.

Analogous to the construction of a priority queue in Figure 5, we show a construction of an $N$-to-1 priority queue in Figure 6. To operate such a queue, we view every empty time slot in an input link as a fictitious packet. Fictitious packets are assigned with priorities lower than any real packets. Furthermore, the priorities among fictitious packets are ranked according to their arrival times and their arriving links. By so doing, there is an arriving packet at every input link in every time slot, and there is a total order among all the packets. As such, one can easily map an $N$-to-1 priority queue to an $N$-to-1 *complementary* priority queue with the complementary priority departure property, i.e., the $N$ lowest priority packets are sent out for $c(t) = 0$, and the $N-1$ lowest priority packets and the highest priority packet are sent out for $c(t) = 1$.

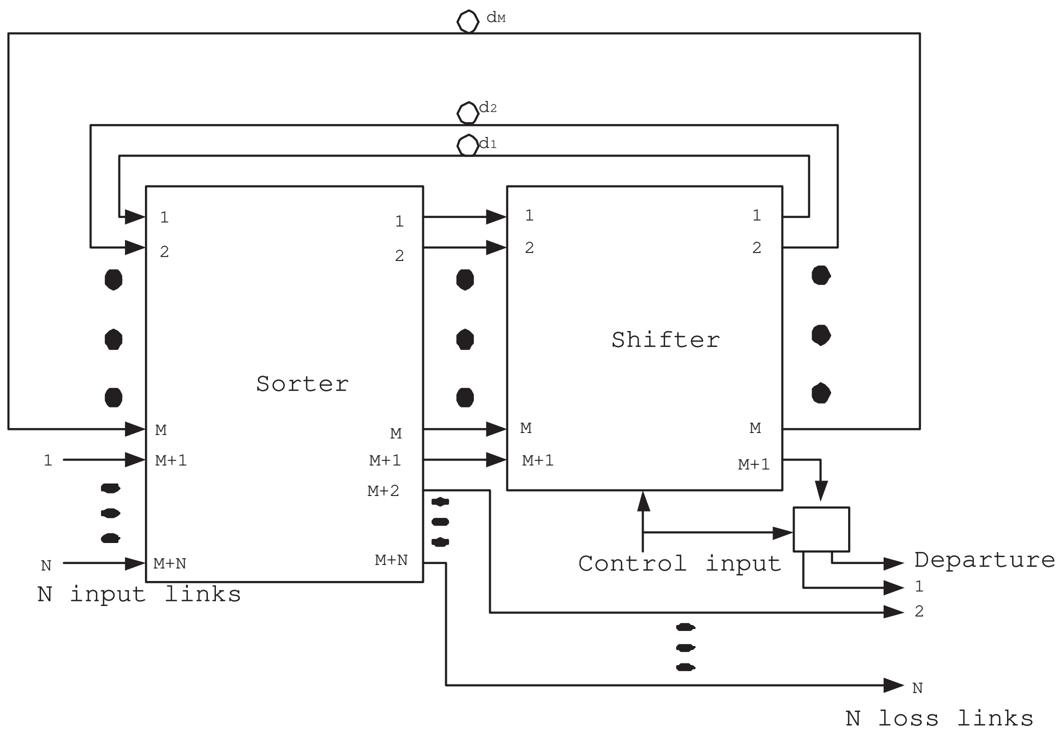


Fig. 6. A construction of an $N$-to-1 priority queue with buffer $\sum_{i=1}^{M} d_i$.

As the construction in Figure 5, the $(M+N) \times (M+N)$ crossbar switch on the left hand side acts as a sorter that sorts the packets at the $M+N$ inputs in the order of their priorities. The first $M+1$ outputs of the sorter are connected to the $(M+1) \times (M+1)$ crossbar switch on the right hand side that acts as a shifter. The remaining $N-1$ outputs of the sorter are connected to the $N-1$ loss links, indexed from $2, 3, \ldots, N$. As before, the shifter only has two connection patterns. Its connection pattern is realized by the $(M+1) \times (M+1)$ identity matrix for $c(t) = 0$ and by the $(M+1) \times (M+1)$ circular-shift matrix for $c(t) = 1$. The $M$ outputs of the shifter, indexed from $i = 1, \ldots, M$ are connected back to the the corresponding $M$ inputs of the sorter via $M$ fiber delay lines with delays $d_i$, $i = 1, \ldots, M$. The $(M+1)^{th}$ output from the shifter is connected to a $1 \times 2$ switch. When $c(t) = 1$, the $(M+1)^{th}$ output from the shifter is connected to the departure link via the $1 \times 2$ switch. On the other hand, it is connected to the first loss link when $c(t) = 0$.

As there might be $N$ lowest priority packets sent out from the corresponding $N$-to-1 complementary priority queue in a time slot, the key insight for the construction in Figure 5 can be modified as follows:

(A2) All the packets stored in all the fiber delay lines in Figure 6 cannot be either the packet with the highest priority or one of the $N$ lowest priority packets until they appear at the inputs of the sorter.

In the following, we provide a condition for (A2) to hold. Under such a condition, we can then emulate an $N$-to-1 priority queue. The proof is almost the same as that for Proposition 4 and thus is omitted.

**Proposition 5** *Suppose that (A2) hold at time 0 and* $0 < d_i \le \min[i, \lfloor \frac{M-i}{N} \rfloor + 1]$ *for all* $i = 1, 2, \ldots, M$, *then the construction in Figure 6 is an* $N$*-to-1 priority queue with buffer* $\sum_{i=1}^{M} d_i$.

As commented before, the two switches in Figure 6 can be replaced by one $(M+N) \times (M+N)$ crossbar switch. Suppose that $M = N(k-1)+k+r$, where $k \ge 1$ and $0 \le r \le N$, then the maximum buffer size that can be achieved by Proposition 5 is to set $d_i = i$ for $1 \le i \le k-1$, $d_i = k$ for $k \le i \le k+r$, and $d_i = k-j$ for $k+r+(j-1)N+1 \le i \le k+r+jN$ and $1 \le j \le k-1$. Therefore, we can construct an $N$-to-1 priority queue with buffer $B = \frac{k(k-1)}{2}(N+1) + (r+1)k$ using a single $((N+1)k+r) \times ((N+1)k+r)$ crossbar switch. Finally, we note that one can further extend our construction to priority queues with multiple inputs and multiple outputs.

## V. Conclusions

In this short paper, we provided a simple proof for the constructions of priority queues. We derived a general result that recovered the result in [10] as a special case. Further, we extended our constructions to priority queues with multiple inputs.

We note that the complexity of our construction is still much larger than the lower bound derived in [10]. For certain types of priority queues, one can push the complexity lower. For example, consider a priority queue that only has $K$ priority classes of packets. Within each class, packets are served in the FIFO order. Then one can construct such a priority queue with $K$ FIFO queues. As a FIFO queue with buffer $B$ can be constructed with $O(\log B)$ $2 \times 2$ crossbar switches (see e.g., [8]), the complexity (in terms of the number of $2 \times 2$ switches) for the construction of such a priority queue is then $O(K \log B)$.

## References


[1] M. J. Karol, “Shared-memory optical packet (ATM) switch,” SPIE vol. 2024 Multigigabit Fiber Communications Systems, pp. 212–222, 1993.

[2] I. Chlamtac, A. Fumagalli, L. G. Kazovsky, P. Melman, W. H. Nelson, P. Poggiolini, M. Cerisola, A. N. M. M. Choudhury, T. K. Fong, R. T. Hofmeister, C. L. Lu, A. Mekkittikul, D. J. M. Sabido IX, C. J. Suh and E. W. M. Wong, “Cord: contention resolution by delay lines,” *IEEE Journal on Selected Areas in Communications*, vol. 14, pp. 1014–1029, 1996.

[3] R. L. Cruz and J. T. Tsai, “COD: alternative architectures for high speed packet switching,” *IEEE/ACM Transactions on Networking*, vol. 4, pp. 11–20, February 1996.

[4] D. K. Hunter, D. Cotter, R. B. Ahmad, D. Cornwell, T. H. Gilfedder, P. J. Legg and I. Andonovic, “$2 \times 2$ buffered switch fabrics for traffic routing, merging and shaping in photonic cell networks,” *IEEE Journal of Lightwave Technology*, vol. 15, pp. 86–101, 1997.

[5] C.-S. Chang, D.-S. Lee and C.-K. Tu, “Recursive construction of FIFO optical multiplexers with switched delay lines,” *IEEE Transactions on Information Theory,* vol. 50, pp. 3221–3233, 2004.

[6] C.-S. Chang, D.-S. Lee and C.-K. Tu, ”Using switched delay lines for exact emulation of FIFO multiplexers with variable length bursts,” to appear in *IEEE Journal on Selected Areas in Communications*. Conference version in *Proceedings of IEEE INFOCOM*, 2003.

[7] C.-C. Chou, C.-S. Chang, D.-S. Lee and J. Cheng, “A necessary and sufficient condition for the construction of 2-to-1 optical FIFO multiplexers by a single crossbar switch and fiber delay lines,” submitted to *IEEE Transactions on Information Theory*.

[8] C.-S. Chang, Y.-T. Chen, and D.-S. Lee, “Construction of optical FIFO queues,” submitted to *IEEE Transactions on Information Theory*.

[9] C.-S. Chang, Y.-T. Chen, J. Cheng, and D.-S. Lee, “Multistage constructions of linear compressors, non-overtaking delay lines, and flexible delay lines,” to appear in *Proceedings of IEEE INFOCOM 2006*.

[10] A. D. Sarwate and V. Anantharam, “Exact emulation of a priority queue with a switch and delay lines,” to appear in *Queueing Systems*.