IEICE TRANS. COMMUN., VOL.E85–B, NO.9 SEPTEMBER 2002

PAPER

Credit-Based Scheduling Algorithms for Input Queued Switch

Jinhui LI[†*], *Nonmember and* **Nirwan ANSARI**[†], *Regular Member*

SUMMARY The input queued (IQ) switching architecture is becoming an attractive alternative for high-speed switches owing to its scalability. In this paper, three new algorithms, referred to as the maximum credit first (MCF), enhanced MCF (EMCF), and iterative MCF (IMCF) algorithms, are introduced. Simulations show that both MCF and IMCF have similar performance as the Birkhoff-von Neumann decomposition (BVND) algorithm, which can provide cell delay bound and 100% throughput, with lower off-line computational and on-line memory complexity. Simulations also show the fairness of MCF is much better than that of BVND. Theoretic analysis shows that the EMCF algorithm has a better performance than MCF in terms of throughput and cell delay with the same complexity level as MCF. Simulation results indicate the EMCF algorithm has much lower average cell delay and delay variance as compared to the BVND algorithm.

key words: *input queued switch, scheduling algorithm*

1. Introduction

There are two basic types of switching architectures: output queued (OQ) and input queued (IQ) switching architecture. When a packet arrives at an OQ switch, it is queued in its output queue. The packet will stay in the output queue until it is transmitted from the switch, and thus 100% throughput can be achieved in OQ switches. OQ switches can also provide quality of service (QoS) guarantees by using scheduling mechanisms [1] such as weighted fair queueing (WFQ) [2] (or packetized generalized processor sharing (PGPS) [3]) and worst-case fair weighted fair queueing (WF^2Q) [4]. The drawback of OQ switches is that in the worst case the fabric of an $N \times N$ OQ switch must run N times as fast as its line rate, i.e., the speedup of an OQ switch is N. On the other hand, when a packet arrives at an IQ switch, it is placed in its input queue until it can be scheduled across the fabric. The packet can be transmitted out of the IQ switch immediately when it arrives the output port. The fabric of an IQ switch needs only run as fast as the line rate, i.e., the speedup of an IQ switch is 1.

In high-speed networks, fabric and memories with a bandwidth operated at N times the line rate may be infeasible. Thus, the IQ switching architecture has been adopted for high-speed switch implementation owing to its scalability. One of the major problems with the IQ switching architecture is the head-of-line (HOL) blocking: the HOL cell, which cannot be forwarded because of the output contention, can block the output-contention-free cells in the same queue when FIFO is used. The HOL blocking limits the throughput of the IQ switch using a single FIFO queue in each input to approximately $2 - \sqrt{2} \approx 58.6\%$ under *i.i.d.* Bernoulli traffic when N is large [5]. Under bursty traffic, it is even worse: the maximum throughput of such a switch decreases monotonically with the burstiness of traffic and reaches 50% when the burstiness is large [6]. Stationary blocking is another problem for FIFO IQ switches: the total throughput of the switch can be as small as the throughput of a single link under certain periodic traffic even when N is very large [7].

Previous research [8][9][10] shows that the HOL blocking can be completely eliminated in IQ switches by adopting virtual output queueing (VOQ), in which multiple VOQs directed to different outputs are maintained at each input. Therefore, the throughput of an IQ switch can be increased to 100% under all admissible independent traffic by using well-designed scheduling algorithms such as the longest queue first (LQF) [9][11] algorithm and the oldest cell first (OCF) [10][11] algorithm. Though 100% throughput can be achieved using IQ switches, other QoS features such as bandwidth and cell delay still cannot be guaranteed using the above algorithms.

Another approach to reduce the HOL blocking is to increase the speed of the fabric. The ratio of bandwidth between the fabric and input link is defined as the speedup. When the speedup is larger than 1 and smaller than N, buffers are required at the outputs as well as inputs. This switching architecture with both input and output buffering is called the combined input output queueing (CIOQ) switch. Enormous efforts have been made on providing QoS guarantees with the CIOQ switch. Recently, Chuang *et al.* [12] proved that a CIOQ switch using the stable matching algorithm [13] and critical cells first (CCF) insertion policy with a speedup of two can exactly mimic an OQ switch that uses the push-in first-out (PIFO) queueing policy.

Other efforts have also been made to achieve QoS guarantees by IQ switch without speedup. Chang *et*

Manuscript received July 16, 2001.
Manuscript revised January 18, 2002.

[†]The authors are with the Advanced Networking Laboratory, Department of Electrical and Computer Engineering, New Jersey Institute of Technology, University Heights, Newark, NJ 07102, USA.
[*]Presently, the author is with Agere Systems.

al. [14][15] proposed the Birkhoff-von Neumann decomposition (BVND) algorithm based on a decomposition result by Birkhoff and von Neumann for a doubly stochastic matrix. This algorithm can provide 100% throughput for all non-uniform traffic. Furthermore, if the traffic is (σ, r)-upper constrained, cell delay can be deterministically guaranteed using this algorithm. The problem of this algorithm is that the off-line computational complexity is too high: $O(N^{4.5})$. If the assigned rates of sessions change frequently, which more likely occurs in a large switch, the algorithm will be impractical. In this paper, we proposed three new algorithms, maximum credit first (MCF), enhanced MCF (EMCF), and iterative MCF (IMCF) algorithms, which have lower off-line complexity and similar performance as the BVND algorithm, and are more realizable.

The rest of the paper is organized as follows. In Sect. 2, we describe our switch model and algorithms. In Sect. 3, we present the discussion and simulation results of proposed algorithms. Concluding remarks are given in Sect. 4.

2. Our Switch Model and Algorithms

Taking into consideration of an $N \times N$ input queued switch without speedup consisting of N inputs, N outputs, and a non-blocking switch fabric such as the crossbar. The packets, which may have variable length, are broken into fixed-length cells when they arrive in the inputs. After the cells cross the fabric, they are reassembled to the original variable length packets. A cell slot is defined as the time required to transmit a cell with the line rate.

The basic objective of scheduling an IQ switch is to find a contention-free match which is equivalent to solving a bipartite graph matching problem as shown in Fig. 1(a). Each vertex on the left side represents an input and that on the right side represents an output. There is an edge between every input vertex i and every output vertex j. Associated with each edge is a weight, $w_{i,j}$, which is defined differently by different algorithms. In Fig. 1(a), edges with a weight of 0 are omitted. The scheduler selects a match between the inputs and outputs with the constraints of unique pairing, i.e., at most one input can be matched to each output, and vice versa. Let $\mathbf{S} = (s_{i,j})$ be the matching matrix, which indicates the match between inputs and outputs. If input i and output j are matched, then $s_{i,j} = 1$; otherwise, $s_{i,j} = 0$. At the end of the cell slot, a cell is transmitted from input i to output j if $s_{i,j} = 1$ and $Q_{i,j}$ is not empty. A maximum weighted matching algorithm computes a match that can maximize the aggregate weight. Figure 1(b) is the maximum weighted matching solution of Fig. 1(a).

2.1 The Maximum Credit First Algorithm

Suppose the rate assigned to the traffic from input i to output j is $r_{i,j}$, which is also the arrival rate of $Q_{i,j}$. The traffic is admissible if and only if the following inequalities are satisfied:

$$\sum_{j=0}^{N-1} r_{i,j} \le 1, \forall i, \tag{1}$$

$$\sum_{i=0}^{N-1} r_{i,j} \le 1, \forall j. \tag{2}$$

$\mathbf{R} = (r_{i,j})$ that satisfies Eqs. (1) and (2) is a doubly substochastic matrix. For any doubly substochastic matrix $\mathbf{R}$, there exists [14][16] a doubly stochastic matrix $\tilde{\mathbf{R}} = (\tilde{r}_{i,j})$ such that $r_{i,j} \le \tilde{r}_{i,j}, \forall i, j$. Matrix $\tilde{\mathbf{R}}$ is a doubly stochastic matrix, if it satisfies

$$\sum_{j=0}^{N-1} \tilde{r}_{i,j} = 1, \forall i, \tag{3}$$

and

$$\sum_{i=0}^{N-1} \tilde{r}_{i,j} = 1, \forall j. \tag{4}$$

The doubly stochastic matrix can be constructed from the doubly substochastic matrix by the weighted rate filling algorithm (WRFA) [17] with the complexity of $O(N^2)$:

Weighted Rate Filling Algorithm (WRFA)

1. Define $p_i = 1 - \sum_{j=0}^{N-1} r_{i,j}$. Calculate p_i for all i.
2. Define $q_j = 1 - \sum_{i=0}^{N-1} r_{i,j}$. Calculate q_j for all j.
3. Calculate $\Delta = N - \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} r_{i,j}$.
4. Let $\tilde{r}_{i,j} = r_{i,j} + \frac{p_i q_j}{\Delta}$.

Theorem 1: Matrix $\tilde{\mathbf{R}} = (\tilde{r}_{i,j})$ constructed from the doubly substochastic matrix $\mathbf{R} = (r_{i,j})$ is a doubly stochastic matrix.

Proof: Since $\mathbf{R} = (r_{i,j})$ is a doubly substochastic matrix, we have $p_i \ge 0, \forall i$, $q_j \ge 0, \forall j$, and $\Delta \ge 0$. Thus, $\tilde{r}_{i,j} = r_{i,j} + \frac{p_i q_j}{\Delta} \ge 0, \forall i, j$. For any j, we have

$$\sum_{i=0}^{N-1} \tilde{r}_{i,j} = \sum_{i=0}^{N-1} r_{i,j} + \sum_{i=0}^{N-1} (\frac{p_i q_j}{\Delta}) = 1 - q_j +$$

$$\frac{\sum_i (1 - \sum_j r_{i,j}) q_j}{\Delta} = 1 - q_j + \frac{(N - \sum_{i,j} r_{i,j}) q_j}{\Delta}$$

$$= \frac{\Delta - q_i \Delta + N q_i - q_i (N - \Delta)}{\Delta} = 1.$$

For any i, we also have $\sum_{j=0}^{N-1} \tilde{r}_{i,j} = 1$. Thus, matrix

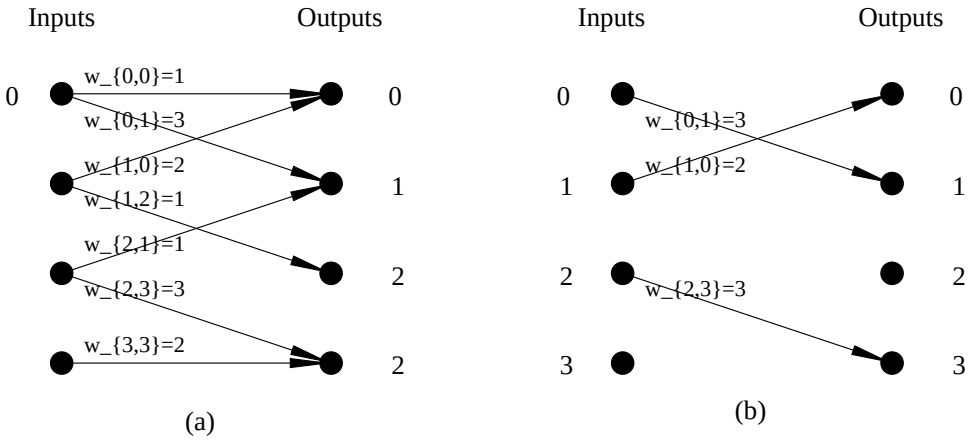


Fig. 1 An example of the bipartite graph matching problem: (a) the request graph and (b) a maximum weighted match.

$\tilde{\mathbf{R}} = (\tilde{r}_{i,j})$ is a doubly stochastic matrix. □

Assume every $Q_{i,j}$ has a credit $c_{i,j}$ which is a real variable that has the initial value of 0. At the beginning of every cell slot, $c_{i,j}$ increases by $\tilde{r}_{i,j}$. Then the scheduler selects the match according to the current credits of VOQs. In this paper, we only consider permutation matrices as the service matrix $\mathbf{S}$, implying that there are always exactly N pairs in every match. From all the permutation matrices, the maximum credit first (MCF) algorithm selects the one that can maximize the aggregate credit, i.e.,

$$\arg\max_{S}[\sum_{i,j} s_{i,j}(n)c_{i,j}(n)],$$

where $\sum_j s_{i,j}(n) = \sum_i s_{i,j}(n) = 1, \forall i,j$. The service matrix $\mathbf{S}$ can be found by the maximum weighted matching algorithm which has a computational complexity of $O(N^3)$ [18]. The cells in VOQs are transmitted across the fabric according to their corresponding values in service matrix $\mathbf{S}$: If $s_{i,j}$ equals to 1 and $Q_{i,j}$ is not empty, then the HOL cell of $Q_{i,j}$ will be transmitted to output j. Meanwhile, $c_{i,j}$ decreases by $s_{i,j}$ for all i,j. Thus, $c_{i,j}(n)$, the credit of $Q_{i,j}$ at the end of cell slot n, is

$$c_{i,j}(n) = c_{i,j}(n-1) - s_{i,j}(n) + \tilde{r}_{i,j}, \tag{5}$$

where $c_{i,j}(n-1)$ is the credit of $Q_{i,j}$ at the end of cell slot $n-1$.

2.2 The Enhanced Maximum Credit First Algorithm

The BVND and MCF algorithms are not efficient enough because they pay no attention to the current occupancy of VOQs. For example, connection requests at the current cell slot is shown in Fig 2(a), where the non-empty VOQs are filled with crosses. Suppose nonzero elements of the permutation matrix selected by the BVND or MCF at the current cell slot are represented by circles in Fig. 2(a). Among all the VOQs selected by $\mathbf{P}$, only VOQ $Q_{0,1}$ is non-empty which is shown by thick border box in Fig. 2(a). Hence, the BVND or MCF algorithm can schedule only one cell in the current cell slot. However, two more VOQs, such as $Q_{1,2}$ and $Q_{2,3}$, can actually send cells across the fabric in the current cell slot without removing cells scheduled by the BVND or MCF algorithm.

Based on the above observation, the enhanced MCF (EMCF) algorithm matches the inputs and outputs, which are not matched by the MCF algorithm, and attempts to make a maximal match in every cell slot using a method similar to the wrapped wave front arbiter (WWFA) [19] and wave front BVND (WFBVND) algorithm [17]. EMCF divides a cell slot into N phases. Assume $\mathbf{P} = (p_{i,j})$ is the permutation matrix selected by the MCF algorithm in the current cell slot. In the lth phase, where $0 \le l \le N-1$, EMCF calculates matrix $\mathbf{V}_l = (v^l_{i,j})$, where $v^l_{i,j} = p_{(i+l)modN,j}$. During the lth phase, EMCF checks VOQs corresponding to the non-zero elements of $\mathbf{V}_l$, and adds the non-empty VOQs in the match if both its input and output are unmatched. EMCF updates credits of VOQs using the following equation:

$$c_{i,j}(n) = c_{i,j}(n-1) - p_{i,j}(n) + \tilde{r}_{i,j}, \tag{6}$$

where $\mathbf{P} = (p_{i,j})$ is the permutation matrix selected by the MCF algorithm.

Figure 2(a)-(d) shows an example, where the VOQs filled with crosses indicate the non-empty VOQs, the VOQs filled with circles indicate that corresponding elements of V_l are $1'$s, and the VOQs with thick border indicate they are scheduled to transmit cells. The complexity of MCF and the filling process are $O(N^3)$ and $O(N^2)$, respectively. Thus, the complexity of EMCF is $O(N^3)$, which is at the same level as MCF.

Theorem 2: If the exactly same traffic is fed into switch $\mathcal{M}$, the IQ switch using the MCF algorithm, and $\mathcal{N}$, the IQ switch using the EMCF algorithm, concurrently and no cell is dropped, then $L^{\mathcal{N}}_{i,j}(n)$, the queue

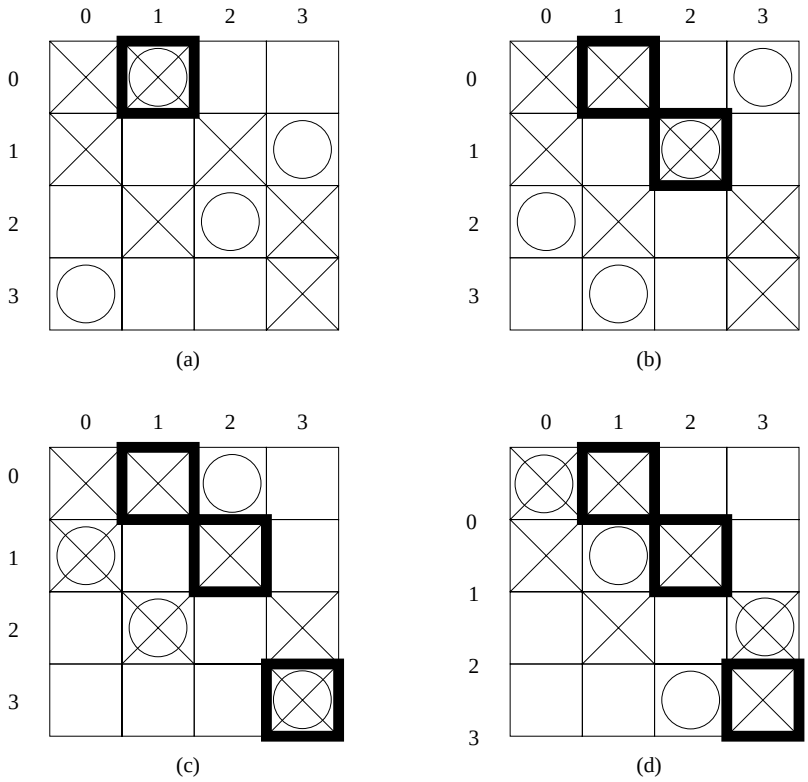


Fig. 2 EMCF algorithm, (a) phase 0, (b) phase 1, (c) phase 2, and (d) phase 3.

length of $Q_{i,j}$ in switch $\mathcal{N}$ at cell slot n, is always less than or equal to $L^{\mathcal{M}}_{i,j}(n)$ for any i, j, and n.

Proof: Consider any i, j, and n, if $L^{\mathcal{N}}_{i,j}(n) = 0$, then the theorem is proved, because $L^{\mathcal{M}}_{i,j}(n) \geq 0$. If $L^{\mathcal{N}}_{i,j}(n) > 0$, let $n_0 < n$ be the largest number such that $L^{\mathcal{N}}_{i,j}(n_0) = 0$. That is, from cell slot $n_0 + 1$ to cell slot n, $Q^{\mathcal{N}}_{i,j}$ is constantly backlogged. Denote $T^{\mathcal{M}}_{i,j}(n_0, n)_{backlog}$ as the number of cells dequeued from $Q^{\mathcal{M}}_{i,j}$ between cell slot $n_0 + 1$ and n inclusively when $Q^{\mathcal{M}}_{i,j}$ is constantly backlogged during $(n_0, n]$.

Let $T^{\mathcal{M}}_{i,j}(n)$ be the cumulative number of cells dequeued from $Q^{\mathcal{M}}_{i,j}$ by the end of cell slot n, and $T^{\mathcal{N}}_{i,j}(n)$ be that of $Q^{\mathcal{N}}_{i,j}$. Define $T_{i,j}(n_2, n_1) = T_{i,j}(n_2) - T_{i,j}(n_1)$. When both $Q^{\mathcal{M}}_{i,j}$ and $Q^{\mathcal{N}}_{i,j}$ are not empty at certain cell slot, if a cell is dequeued from $Q^{\mathcal{M}}_{i,j}$, then a cell must be dequeued from $Q^{\mathcal{N}}_{i,j}$ according to the definition of EMCF. Thus, we know that $T^{\mathcal{N}}_{i,j}(n_0, n) \geq T^{\mathcal{M}}_{i,j}(n_0, n)_{backlog}$. Thus, we have $T^{\mathcal{N}}_{i,j}(n_0, n) \geq T^{\mathcal{M}}_{i,j}(n_0, n)$, because $T^{\mathcal{M}}_{i,j}(n_0, n)_{backlog} \geq T^{\mathcal{M}}_{i,j}(n_0, n)$. Since $L^{\mathcal{N}}_{i,j}(n_0) = 0$, $L^{\mathcal{N}}_{i,j}(n_0) \leq L^{\mathcal{M}}_{i,j}(n_0)$. From cell slot $n_0 + 1$ to n, the same number of cells are enqueued into $Q^{\mathcal{M}}_{i,j}$ and $Q^{\mathcal{N}}_{i,j}$, but more or the same number of cells are dequeued from $Q_{i,j}$. Thus $L^{\mathcal{N}}_{i,j}(n) \leq L^{\mathcal{M}}_{i,j}(n)$ for any i, j, and n. □

Theorem 3: Assume all the VOQs in switch $\mathcal{M}$ and $\mathcal{N}$ are FIFOs. If the exactly same traffic is fed into switch $\mathcal{M}$ and $\mathcal{N}$ concurrently and no cell is dropped, then $D^{\mathcal{N}}_{\mathbf{c}}$, the delay time of cell $\mathbf{c}$ in switch $\mathcal{N}$, is always less than or equal to $D^{\mathcal{M}}_{\mathbf{c}}$ for any cell $\mathbf{c}$.

Proof: If a cell c which is directed to output j arrives in input i at cell slot n, then both $Q^{\mathcal{M}}_{i,j}$ and $Q^{\mathcal{N}}_{i,j}$ will be backlogged until c is scheduled. At cell slot n, $L^{\mathcal{M}}_{i,j}(n) \geq L^{\mathcal{N}}_{i,j}(n)$. Assume cell c departs switch E at cell slot n_1, then $T^{\mathcal{N}}_{i,j}(n, n_1) \geq T^{\mathcal{M}}_{i,j}(n, n_1)$. Since the arrival of B and E are identical, cell $\mathbf{c}$ cannot leave switch B before cell slot n_1, if all the VOQs are FIFOs. So, $D^{\mathcal{N}}_{\mathbf{c}} \leq D^{\mathcal{M}}_{\mathbf{c}}$ for any cell $\mathbf{c}$. □

Theorem 2 and 3 prove that the performance of EMCF is better than that of MCF in terms of cell delay and buffer length.

2.3 The Iterative Maximum Credit First Algorithm

Owing to the $O(N^3)$ complexity, MCF and EMCF algorithms are difficult to implement in high speed networks, so we propose an iterative approximation of MCF algorithm: iterative maximum credit first (IMCF) algorithm. IMCF performs the following three steps in each iteration:

1. Request: Each unmatched input sends a request to every output.
2. Grant: If an unmatched output receives any requests, it chooses the one with the largest credit. Ties are broken randomly.

3. Accept: If an input receives any grants, it chooses the one with the largest credit. Ties are broken randomly.

IMCF stops when there is no unmatched input and output. It converges in at most N iterations and the service matrix $\mathbf{S}$ selected by IMCF is always a permutation matrix.

Property 1: At the end of any cell slot, credits of an IQ switch using MCF, EMCF, or IMCF satisfy the following equations:

$$\begin{cases} \sum_{j=1}^{N} c_{i,j} = 0, & \forall i \\ \sum_{i=1}^{N} c_{i,j} = 0, & \forall j \\ \sum_{i,j} c_{i,j} = 0 \end{cases} . \tag{7}$$

Property 2: In any cell slot just before the service matrix is calculated, credits of an IQ switch using MCF, EMCF, or IMCF satisfy the following equations:

$$\begin{cases} \sum_{j=1}^{N} c_{i,j} = 1, & \forall i \\ \sum_{i=1}^{N} c_{i,j} = 1, & \forall j \\ \sum_{i,j} c_{i,j} = N \end{cases} . \tag{8}$$

Property 3: If two cells arrive from the same input and come to the same output of an IQ crossbar switch that uses the MCF, EMCF, or IMCF algorithm, then the cell that arrives early will depart the switch early.

Property 3 implies that if the MCF, EMCF, or IMCF algorithm is used to handle variable-length packets, reordering is not necessary at the output side.

3. Discussion and Simulations

Let $T_{i,j}(n)$ be the cumulative number of cell slots that are assigned to $Q_{i,j}$ of an MCF switch by cell slot n. Then the credit of $Q_{i,j}$ is

$$c_{i,j}(n) = \tilde{r}_{i,j} \cdot n - T_{i,j}(n). \tag{9}$$

For any $n_2 \geq n_1$,

$$T_{i,j}(n_2) - T_{i,j}(n_1) = \tilde{r}_{i,j}(n_2 - n_1) - c_{i,j}(n_2) + c_{i,j}(n_1). \tag{10}$$

If we assume the lower bound and upper bound of the credit are c^- and c^+, respectively, and let $\Delta_c = c^+ - c^-$, then

$$\begin{aligned} \tilde{r}_{i,j}(n_2 - n_1) - \Delta_c &\leq T_{i,j}(n_2) - T_{i,j}(n_1) \\ &\leq \tilde{r}_{i,j}(n_2 - n_1) + \Delta_c. \end{aligned} \tag{11}$$

Eq. (11) implies that if $A_{i,j}$, the traffic from input i to output j, conforms to $(\sigma_{i,j}, r_{i,j})$, i.e.,

$$A_{i,j}(n_2) - A_{i,j}(n_1) \leq r_{i,j}(n_2 - n_1) + \sigma_{i,j}, \tag{12}$$

then the cell delay from input i to output j is bounded by $\lceil (\sigma_{i,j} + \Delta_c)/r_{i,j} \rceil$.

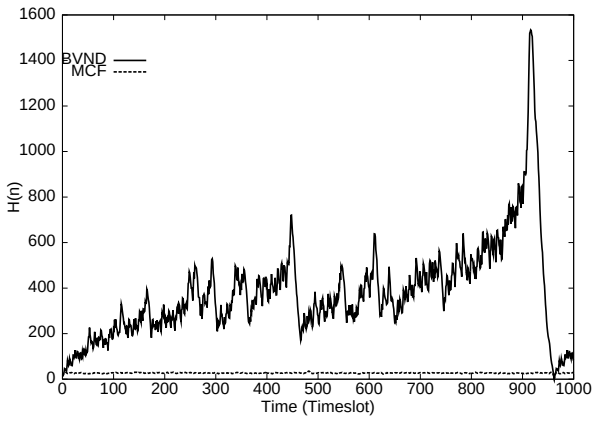


Fig. 3 $H(n)$ of BVND and MCF under unbalanced traffic reservation.

Chang *et al.* [14] show if Eq. (1) and (2) are satisfied, then the BVND algorithm can guarantee

$$T_{i,j}(n_2) - T_{i,j}(n_1) \geq r_{i,j}(n_2 - n_1) - u_{i,j}, \tag{13}$$

for all i, j, $n_2 \geq n_1$, where $u_{i,j} \leq U = N^2 - 2N + 2$.

Comparing Eq. (11) with Eq. (13), we can say if the Δ_c of MCF is comparable with $U = N^2 - 2N + 2$, then MCF is as good as the BVND algorithm in terms of QoS guarantees. Simulation results of c^+, c^-, and Δ_c are listed in Table 1.Simulations are made under various non-uniform rate reservations on input queued switches with $N = 16$, $N = 32$, and $N = 64$. Table 1 shows that the Δ_c of MCF is much smaller than U.

Both BVND and MCF try to keep the credits of VOQs as close to 0 as possible to ensure the fairness, where the flows reserved with higher rates should be assigned more cell slots to transmit cells. Thus $H(n) = \sum_{i=0,j=0}^{N-1,N-1} c_{i,j}^2(n)$ can be used as the measurement of the fairness. For an ideal scheduling algorithm in the fluid model, $H(n)$ should be equal to 0 at any cell slot, which is not possible in the packetized case. In practice, an algorithm with better fairness should have a smaller $H(n)$. We have

$$\begin{aligned} H(n+1) - H(n) &= N - \sum \tilde{r}_{i,j}^2 \\ &+ 2 \sum c_{i,j}(n) \tilde{r}_{i,j} - 2 \sum c'_{i,j}(n) s_{i,j}, \end{aligned}$$

where $c'_{i,j}(n) = c_{i,j}(n) + \tilde{r}_{i,j}$. Since MCF selects the matching matrix $\mathbf{S}$ which has the maximum $\sum c'_{i,j}(n) s_{i,j}$, MCF can minimize $H(n)$. The comparison of BVND and MCF under an unbalanced traffic reservation, which is generated randomly, for a 16×16 switch is shown in Fig. 3. In this example, the maximum $H(n)$ of BVND and MCF are 1550 and 30 respectively, which implies MCF has much better fairness than BVND.

Figures 4 and 5 show the distribution of the percentage of cells which experience various delays over a

Table 1 Maximum and minimum credit of MCF

N	16			32			64		
U	226			962			3970		
Algorithm	c^+	c^-	Δ_c	c^+	c^-	Δ_c	c^+	c^-	Δ_c
MCF	1.08	−1.07	2.15	1.05	−0.96	2.01	0.95	−0.98	1.93

Table 2 Computational and memory complexity

Algorithm	off-line computational	on-line computational	on-line memory
BVND	$O(N^{4.5})$	$O(logN)$	$O(N^3logN)$
MCF	$O(N^2)$	$O(N^3)$	$O(N^2)$
EMCF	$O(N^2)$	$O(N^3)$	$O(N^2)$
IMCF	$O(N^2)$	$O(N^2)$	$O(N^2)$

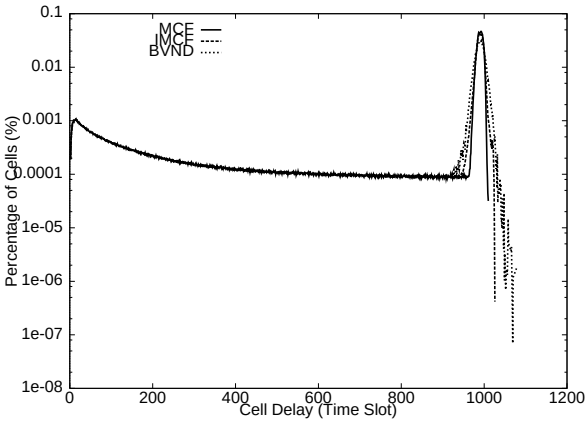


Fig. 4 Cell delay distribution.

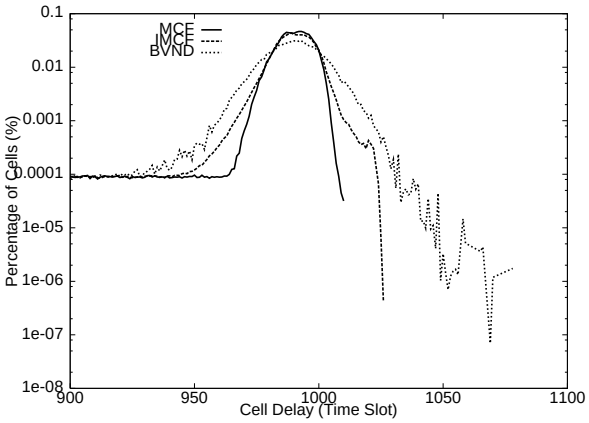


Fig. 5 Cell delay distribution.

16×16 IQ switch using MCF, IMCF, and BVND algorithms under non-uniform traffic with a total traffic load of 90%. Traffic $A_{i,j}$ conforms to $(\sigma_{i,j}, r_{i,j})$ for all i, j, in which $\sigma_{i,j} = 1000 r_{i,j}$. Thus, the delay bound should be $1000 + \lceil u_{i,j}/r_{i,j} \rceil$ and $1000 + \lceil \Delta_c / r_{i,j} \rceil$ cell slots for BVND and MCF, respectively. The simulation result shows that the cell delay bound is less than 1200 cell slots for all the algorithms as expected. Figures 4 and 5 demonstrate that the performance of the BVND algorithm and IMCF are almost identical, and MCF has a tighter bound than the other two algorithms.

When $r_{i,j}$, the reserved rate between input i and output j, changes, MCF and IMCF need to recalculate $\tilde{\mathbf{R}}$. On the other hand, the BVND algorithm not only needs to recalculate $\tilde{\mathbf{R}}$, but also needs to decompose it again. Owing to the high off-line computational complexity, the BVND algorithm will be hard to implement in a dynamic environment. Table 2 compares the computational and memory complexity of the BVND, MCF, EMCF, and IMCF algorithms. It indicates that MCF and IMCF have higher on-line computational complexity, but have lower off-line computational complexity and on-line memory complexity. The BVND algorithm's bottleneck is the off-line computation with the complexity of $O(N^{4.5})$, which needs to be calculated whenever the traffic matrix $\mathbf{R}$ is changed; the MCF algorithm's bottleneck is the on-line computation with the complexity of $\mathrm{O}(N^3)$, which needs to be calculated every cell slot. Assume traffic matrix $\mathbf{R}$ changes every τ cell slots on average. For a rough estimation, we can say that when τ is less than $N^{1.5}$, MCF is easier to be calculated than BVND. When N is 512, τ is about $2ms$ for ATM traffic at a line rate of OC-48. Matrix $\mathbf{R}$ changes much more frequently than its elements. τ is $2ms$ when the duration of a single connection is about $500s$, if we assume there is only one connection from one input to one output. The above calculation implies MCF is easier to be implemented than BVND when the duration of a single connection is smaller than $500s$, when N is 512.

Figure 6 shows the distribution of the percentage of cells which experience various delays over the switch using MCF and EMCF algorithms under unbalanced (σ, r) traffic with an average traffic load of 79%, in which $r_{i,j}$ is selected randomly, and $\sigma_{i,j}$ is set to be $1000 r_{i,j}$. In this example, the traffic was generated with the $on - off$ model and constrained by leaky bucket. This figure demonstrates that EMCF has a smaller delay bound than MCF. Under this traffic condition, the average cell delay of MCF is 850 cell slots. On the other hand, the average cell delay of EMCF is only 51, which is much smaller than MCF.

Figure 7 shows the average cell delay of BVND and EMCF versus the traffic load under *i.i.d.* (σ, r) arrival.

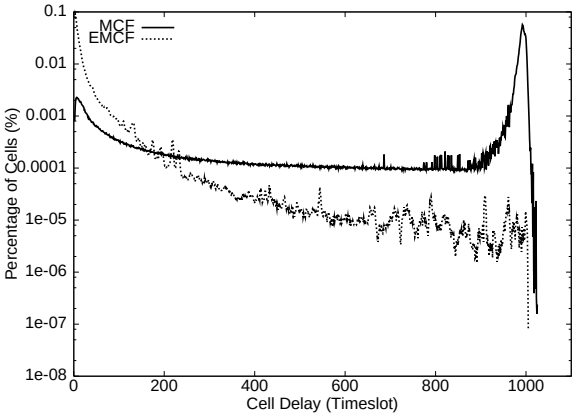


Fig. 6 Cell delay distribution under unbalanced (σ, r) traffic using MCF and EMCF algorithms.

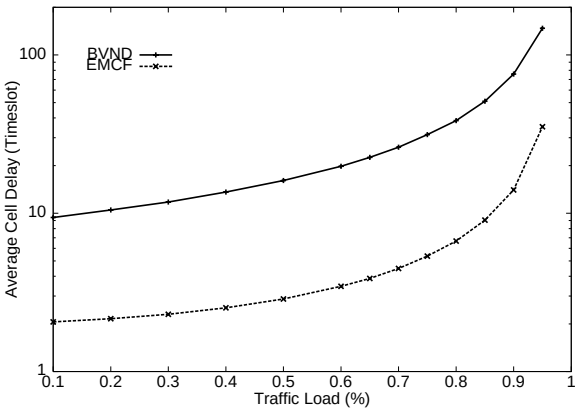


Fig. 7 Average cell delay vs. traffic load under *i.i.d.* (σ, r) traffic using BVND and EMCF algorithms.

It indicates that EMCF reduces the average cell delay significantly. Figure 8 is the variance of cell delay versus the traffic load under the same traffic model as Fig. 7. Comparing with BVND, the variance of cell delay of EMCF is also much less.

4. Conclusions

In this paper, we have proposed MCF, EMCF, and IMCF algorithms, which have the similar delay bound as the BVND algorithm. These new algorithms have less off-line computational and on-line memory complexity, that makes them easier to be implemented under dynamic environments. Furthermore, the MCF algorithm has much better fairness than the BVND algorithm. The EMCF algorithm provides much lower average cell delay and delay variance than that of the BVND algorithm.

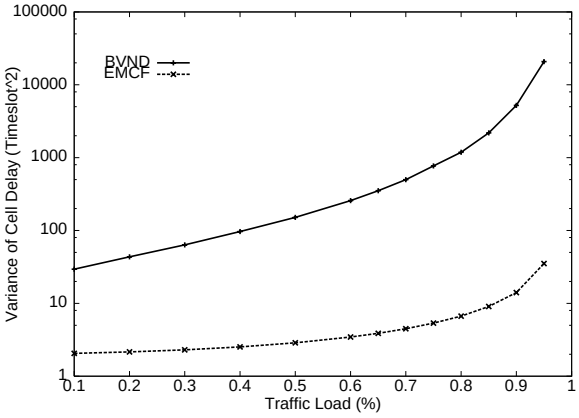


Fig. 8 Variance of cell delay vs. traffic load under *i.i.d.* (σ, r) traffic using BVND and EMCF algorithms.

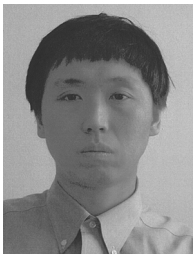
Acknowledgments

This work has been supported in part by the New Jersey Commission on Science and Technology via the New Jersey Center for Wireless Telecommunications, and the New Jersey Commission on Higher Education via the NJI-TOWER project.

References

[1] H. Zhang, "Service disciplines for guaranteed performance service in packet-switching networks," Proc. IEEE, vol.83, no.10, pp. 1374–1396, October 1995.
[2] A. Demers, S. Keshav, and S. Shenkar, "Analysis and simulation of a fair queueing algorithm," Internet. Res. and Exper., vol.1, no.1, pp. 3–26, Sept. 1990.
[3] A.K. Parekh, R.G. Gallager, "A generalized processor sharing approach to flow control in integrated services networks: the single node case," IEEE Trans. on Networking, vol.1, no.3, pp. 344–357, June 1993.
[4] J.C.R. Bennett and H. Zhang, "WF^2Q: worst-case fair weighted fair queueing," Proc. INFOCOM, San Francisco, USA, pp. 120–148, March 1996.
[5] M. Karol, M. Hluchyi, S. Mogan, "Input versus output queueing on a space-division packet switch," IEEE Trans. Commun., vol.COM-35, pp. 1347–1356, Dec. 1987.
[6] S.-Q. Li, "Performance of a nonblocking space-division packet switch with correlated input traffic," Proc. IEEE GlOBECOM, pp. 1754–1763, Dallas, USA, Nov. 1989.
[7] S.-Y. Li, "Theory of periodic contention and its application to packet switching," Proc. INFOCOM, pp. 320–325, New Orleans, USA, March 1988.
[8] T. Anderson, S. Owicki, J. Saxe and C. Thacker, "High speed switch scheduling for local area networks," ACM Trans. on Computer Systems, pp. 319–352, November 1993.
[9] N. Mckeown, V. Anantharam and J. Walrand, "Achieving 100% throughput in an input-queued switch," Proc. IEEE INFOCOM, San Francisco, USA, pp. 296–302, March 1996.
[10] A. Mekkittikul and N. McKeown, "A starvation-free algorithm for achieving 100% throughput in an input-queued switch," Proc. ICCCN, pp. 226–231, Rockville, USA, Oct. 1996.
[11] N. McKeown, A. Mekkittikul, V. Anantharam, and J.

Walrand, "Achieving 100% throughput in an input-queued switch," IEEE Trans. on Commun., vol.47, no.8, pp. 1260–1267, August 1999.
[12] S.-T. Chuang, A. Goel, N. Nckeown, and B. Prabhakar, "Matching output queueing with a combined input/output-queued switch," IEEE J. Select. Areas Commun., vol.17, no.6, pp. 1030–1039, June 1999.
[13] D. Gale and L.S. Shapley, "College admissions and the stability of marriage," American Mathematical Monthly, vol.69, pp. 9–15, 1962.
[14] C.-S. Chang, W.-J. Chen, H.-Y. Huang, "On service guarantees for input buffered crossbar switches: a capacity decomposition approach by Birkhoff and von Neumann," Proc. IWQOS, pp. 79–86, London, England, June 1999.
[15] C.S. Chang, W.J. Chen, and H.Y. Huang, "Birkhoff-von Neumann input buffered crossbar switches," Proc. IEEE INFOCOM, Tel Aviv, Israel, pp. 1614–1623, March 2000.
[16] J. von Neumann, "A certain zero-sum two-person game equivalent to the optimal assignment problem," in Contributions to the Theory of Games, vol.2, pp. 5–12, Princeton University Press, Princeton, USA, 1953.
[17] J. Li and N. Ansari, "QoS guaranteed input queued scheduling algorithms with low delay," Proc. IEEE Workshop on High Performance Switching and Routing, Dallas, USA, pp. 412–414, May 2001.
[18] R.E. Tarjan, Data structures and network algorithms, Society for Industrial and Applied Mathematics, Pennsylvania, 1983.
[19] Y. Tamir and H.C. Chi "Symmetric crossbar arbiters for VLSI communication switches," IEEE Trans. on Parallel and Distributed Systems, vol.4, no.1, pp. 13–27, January 1993.



Jinhui Li received the B.S. Degree in physics and the M.S. degree in electrical engineering from Peking University, Beijing, China, and Ph.D. degree in electrical engineering from New Jersey Institute of Technology, Newark, NJ, USA, in 1991, 1994, and 2001, respectively. He has been a System Architect in Agere Systems, Allentown, PA, USA, since 2001.

Nirwan Ansari received the B.S.E.E. (summa cum laude), M.S.E.E., and Ph.D. from NJIT, University of Michigan, and Purdue University in 1982, 1983, and 1988, respectively. He joined the Department of Electrical and Computer Engineering, NJIT, in 1988, and has been Professor since 1997. He is a technical editor of the IEEE Communications Magazine, was instrumental, while serving as its Chapter Chair, in rejuvenating the North Jersey Chapter of the IEEE Communications Society which received the 1996 Chapter of the Year Award, currently serves as the Chair of the IEEE North Jersey Section, and also serves in the IEEE Region 1 Board of Directors and various IEEE committees. He was the 1998 recipient of the NJIT Excellence Teaching Award in Graduate Instruction, and a 1999 IEEE Region 1 Award. His current research focuses on various aspects of high speed networks. He authored with E.S.H. Hou Computational Intelligence for Optimization (1997, and translated into Chinese in 2000), and edited with B. Yuhas Neural Networks in Telecommunications (1994), both published by Kluwer Academic Publishers.