# Performance Analysis of a Dual Round Robin Matching Switch with Exhaustive Service

Yihan Li, Shivendra Panwar,H. Jonathan Chao

*Abstract*— **Virtual Output Queuing is widely used by fixed-length high-speed switches to overcome head-of-line blocking. This is done by means of matching algorithms. Maximum matching algorithms have good performance, but their implementation complexity is quite high. Maximal matching algorithms need speedup to guarantee good performance. Iterative algorithms (such as PIM and iSLIP) use multiple iterations to converge on a maximal match. The Dual Round-Robin Matching (DRRM) scheme has performance similar to iSLIP and lower implementation complexity. The objective of matching algorithms is to reduce the matching overhead for each time slot. The Exhaustive Service Dual Round-Robin Matching (EDRRM) algorithm amortizes the cost of a match over multiple time slots. While EDRRM suffers from a throughput below 100% for small switch sizes, it is conjectured to achieve an asymptotic 100% throughput under uniform traffic. Simulations show that it achieves high throughput under nonuniform traffic. Its delay performance is not sensitive to traffic burstiness, switch size and packet length. In an EDRRM switch cells belonging to the same packet are transferred to the output continuously, which leads to good packet delay performance and simplifies the implementation of packet reassembly. In this paper we analyze the performance of an EDRRM switch by using an exhaustive service random polling system model. This was used to predict the performance of switches too large to be simulated within a reasonable run time.**

*Index Terms*—**switching, scheduling, Virtual Output Queueing, Dual Round Robin, polling, exhaustive service.**

## I. Introduction

FIXED-LENGTH switching technology is widely accepted as an approach to achieve high switching efficiency for high speed packet switches. Variable-length IP packets are segmented into fixed-length "cells" at inputs and are reassembled at the outputs.

Packet switches based on Input Queuing (IQ) are desirable for high speed switching, since the internal operation speed is only slightly higher than the input line. However, an Input Queuing switch has a critical drawback [1], [2]: the throughput is limited to 58.6% due to the head-of-line (HOL) blocking phenomena. Output Queuing (OQ) switches have optimal delay-throughput performance for all traffic distributions, but the N-times speed-up in the fabric limits the scalability of this architecture.

Virtual Output Queuing (VOQ) is used to overcome the drawbacks and combine the advantages of an Input Queuing switch and an Output Queuing switch. In a VOQ switch, each input maintains $N$ queues, one for each output. By using VOQ, no additional speedup is required and HOL blocking can be eliminated.

Yihan Li is a Ph.D. candidate in the Electrical and Computer Engineering Department, Polytechnic University, Brooklyn, NY 11201, email: yli@photon.poly.edu.

Shivendra Panwar and H. Jonathan Chao are on the faculty of the Electrical and Computer Engineering Department, Polytechnic University, Brooklyn, NY 11201, email: panwar@catt.poly.edu, chao@antioch.poly.edu

This work is supported in part by the New York State Center for Advanced Technology in Telecommunications (CATT), and also in part by the National Science Foundation under grants ANI0081527 and ANI0081357 .

Considerable work has been done on scheduling algorithms for VOQ switches. It has been proved that by using a *maximum weight matching algorithm* 100% throughput can be reached for i.i.d. arrivals (uniform or nonuniform) [3], [4], [5], [6]. But maximum weight matching is not practical to implement in hardware due to its complexity,and may not guarantee fairness and quality of service. A number of practical *maximal matching algorithms* have been proposed [7], [8], [9], but maximal matching algorithm cannot achieve as high a throughput as maximum matching algorithms. Iterative algorithms such as PIM [10] and iSLIP [11], [6], use multiple iterations to converge on a maximal matching.

The Dual Round-Robin Matching (DRRM) switch [12], [13] builds and improves on the ideas incorporated in iSLIP. It has been proven that DRRM can achieve 100% throughput under i.i.d. and uniform traffic [13]. Furthermore, the DRRM scheme provides fairness and prevents starvation. It has lower implementation complexity compared to algorithms with similar performance and is scalable. According to simulation results [13], under uniform bursty traffic, the average delay of a DRRM switch varies approximately linearly with burst length, but under nonuniform traffic the throughput drops below 100%.

Exhaustive service DRRM (EDRRM) [14], a variation of DRRM, improves switching performance under bursty and nonuniform traffic. The implementation of EDRRM and DRRM are comparable with both having lower complexity than iSLIP. According to simulation results, it is conjectured that for an EDRRM switch of large size, throughputs approaching 100% are achievable under uniform traffic. Analysis results in this paper support, though not rigorously prove, this conjecture. Compared to DRRM and iSLIP, EDRRM has higher throughput under nonuniform traffic. The delay of EDRRM is less sensitive to traffic burstiness, and increases much slower with switch size. EDRRM is neither a maximum matching nor a maximal matching algorithm. Unlike any maximum or maximal matching algorithm, which try to find as many matches as possible in each time slot, EDRRM achieves efficiency by looking at the matching overhead over time. In EDRRM the cost in wasted slot times to get a match may be large, but the cost is amortized over a VOQ busy period. We believe that this is a new approach with both theoretical and practical implications.

Usually in a packet switch multiple queues are needed at each Output Reassembly Module (ORM) if cells belonging to different packets are interleaved at the same output [15]. When a cell is transferred through the switch fabric to the output, it is delivered to one of the queues of the ORM. The cells belonging to the same packet will be delivered to the same queue and can only leave the queue until the whole packet is reassembled. The total delay a packet suffers includes the cell delay and the time needed for reassembly. In order to evaluate the variable component of the delay incurred in a packet switch, the packet delay as well as the cell delay of EDRRM is compared to those of DRRM and iSLIP. It shows that under uniform i.i.d. traffic, the packet delay of EDRRM is lower, and not sensitive to

0-7803-7632-3/02/$17.00 ©2002 IEEE

switch size and packet length. At the same time, since all the cells belonging to the same packet are transferred to the output continuously, only one queue is needed in each ORM, which further simplifies the switch implementation.

In this paper we analyze the performance of the EDRRM switch under uniform traffic by using an exhaustive service random polling system model. The analytical result is used to predict the performance of switches too large to be simulated within a reasonable run time.

In section II we briefly review the EDRRM algorithm and its performance. In section III, the EDRRM algorithm is analyzed by modeling it as a polling system.

## II. The Exhaustive service DRRM scheme and its performance

### A. The Exhaustive service DRRM scheme: Motivation, Description, and an Example

In the DRRM scheme [13], each input selects one nonempty VOQ by round robin, and each output accept one of the multiple requests it receives, also in round robin order. When an input and an output are matched, only one cell is transferred from the input to the matched output. After that both the input and the output will increment their pointers by one and in the next time slot this input-output pair will have the lowest matching priority. This behavior is similar to the *limited service policy* [16] in a polling system. In order to improve on DRRM's performance under non-uniform traffic, we modified the DRRM scheme so that whenever an input is matched to an output, all the cells in the corresponding VOQ will be transferred in the following time slots before any other VOQ of the same input can be served. This is called the *exhaustive service policy* [16] in polling systems. We therefore call this the Exhaustive service DRRM (EDRRM) scheme.

In EDRRM, the pointers of inputs and outputs are updated in a different way from DRRM. In a time slot if an input is matched to an output, one cell in the corresponding VOQ will be transferred. After that, if the VOQ becomes empty, the input will update its arbiter pointer to the next location in a fixed order; otherwise, the pointer will remain at the current VOQ so that a request will be sent to the same output in the next time slot. If an input sends a request to an output but gets no grant, the input will update its arbiter pointer to the next location in a fixed order, which is different from DRRM where the input pointer will remain where it is until it gets a grant. The reason for this modification is as follows. In EDRRM if an input cannot get a grant from an output, it means that the output is most likely in a "stable marriage" with another input for all the cells waiting in the VOQ, and the unsuccessful input is likely to wait for a long time to get a grant from this output. It is better for the input to search for another free output than to wait for this busy one. Since an output has no idea if the currently served VOQ will become empty after this service, outputs will not update their arbiter pointers after cell transfer.

A detailed description of the two step EDRRM algorithm follows:

Step 1 : Request. Each input moves its pointer to the first nonempty VOQ in a fixed round-robin order, starting from the current position of the pointer, and sends a request to the output corresponding to the VOQ. The pointer of the input arbiter is incremented by one location beyond the selected output if the request is not granted in Step 2, or if the request is granted and after one cell is served this VOQ becomes empty. Otherwise, the pointer remains at that (nonempty) VOQ.

Step 2 : Grant. If an output receives one or more requests, it chooses the one that appears next in a fixed round-robin schedule starting from the current position of the pointer. The pointer is moved to this position. The output notifies each requesting input whether or not its request was granted. The pointer of the output arbiter remains at the granted input. If there are no requests, the pointer remains where it is.

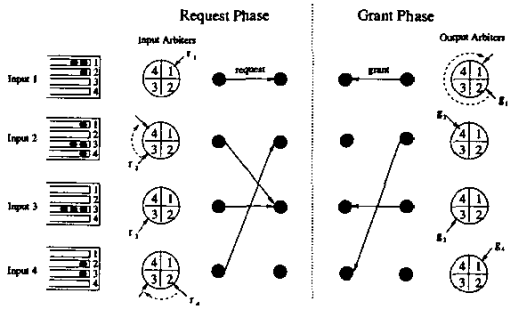


Fig. 1. An example of the EDRRM algorithm

Figure 1 shows an example of the EDRRM arbitration algorithm. $r_1$, $r_2$, $r_3$ and $r_4$ are arbiter pointers for inputs 1, 2, 3 and 4, and $g_1$, $g_2$, $g_3$ and $g_4$ are arbiter pointers for outputs 1, 2, 3 and 4. At the beginning of the time slot $r_1$ points to output 1 while $g_1$ does not point to input 1, which means that in the last time slot input 1 was not matched to output 1, and now input 1 requests output 1 for a new service. Similarly, $r_2$ requests output 3 for a new service. Since $r_3$ points to output 3 and $g_3$ points to input 3, it is possible that in the last time slot input 3 was matched to output 3 and in this time slot output 3 will transfer the next cell from input 3 because the VOQ is not empty. Input 4 and output 2 have a similar situation as input 3 and output 3. In the grant phase, output 1 grants the only request it receives from input 1 and updates $g_1$ to 1, output 2 grants the request from input 4 and output 3 grants the request from input 3. The request from input 2 to output 3 is not granted, so $r_2$ moves to 4. By the end of this time slot, the $1^{st}$ VOQ of input 1 and the $3^{rd}$ VOQ of input 3 are still nonempty so that $r_1$ and $r_2$ are not updated; $r_4$ is updated to 3 because the $2^{nd}$ VOQ of input 4 becomes empty.

The implementation complexity of EDRRM's switching fabric is identical to that of DRRM. Since the operational step and data exchange is limited, the DRRM arbitration mechanism can be implemented in a distributed manner to make the switch simpler and more scalable. The length of each control message in DRRM is only $\frac{1}{N}th$ of that in iSLIP. In [12] it is shown that by using a token-tunneling technique a switch capacity of more than one terabit/sec is achievable with existing electronic technology. The ORM of EDRRM is simpler than that of DRRM. Only one queue, with a buffer size equal to the maximum packet size, is maintained in the ORM of an EDRRM switch since cells belonging to the same packet are served sequentially from a VOQ. Usually, as in DRRM and iSLIP, since cells of different packets are interleaved, $N$ queues are needed in each ORM, one for each input. In the next section, we will show that EDRRM has performance comparable with DRRM and iSLIP under uniform independent traffic, and has better performance under bursty traffic and nonuniform traffic.

### B. Review of the simulated performance of EDRRM

In this subsection we will briefly review the simulation results available in [14] for the EDRRM switch.

According to simulation results the throughput of EDRRM under uniform and i.i.d. traffic is close to, but not quite 100%. For switches with size not larger than 32, the throughput first decreased and then increases with switch size. It is conjectured that for larger $N$ the throughput will approach 100% asymptotically. This conjecture is further supported by the analysis in this paper. While this is certainly a weakness of EDRRM as compared to DRRM and iSLIP, we believe that this is an acceptable tradeoff given its performance advantages under more typical traffic loadings.

The throughput of EDRRM has been simulated under four nonuniform traffic patterns and compared to those of DRRM and iSLIP in [14]. While the throughputs of DRRM and iSLIP drop, EDRRM leads to high throughput, which is always higher or close to 90%. In some extreme traffic pattern, unfairness may occur for an EDRRM switch when one input occupies an output for a long period and cells from other inputs destined to the same output cannot get through. To avoid unfairness, a limit on the maximum number of cells or packets that can be served continuously in a VOQ can be enforced by means of a counter. With a VOQ cell service limit the unfairness can be efficiently avoided and for other traffic patterns the performance of an EDRRM switch does not differ much from the performance of EDRRM with no VOQ cell service limit.

The performance of iSLIP and DRRM are roughly comparable [13]. Under uniform and i.i.d. traffic, the average cell delay of an EDRRM switch under a heavy load is larger than that of a DRRM switch. This is due to the more efficient DRRM scheduling mechanism under uniform, heavy traffic.

Since all the cells arrive within the same burst will be served continuously,EDRRM is not sensitive to bursty traffic. Under uniform and geometrically distributed bursty traffic, with the same average burst length, the average delay of DRRM is much larger than that of EDRRM under heavy load. The average delay of a DRRM switch increases approximately linearly with burst length, which is similar to the behavior of an EDRRM switch under light load. Significantly, under heavy load the average delay of an EDRRM switch does not change much with different average burst lengths and is much smaller than that of a DRRM switch for long burst lengths. Furthermore, the cell delay for EDRRM is less sensitive to switch size than DRRM for bursty traffic. As the switch size increases the average cell delay of a DRRM switch grows rapidly, while the average cell delay of an EDRRM switch grows more slowly.

DRRM and EDRRM are fixed-length switching algorithms. In a fixed-length packet switch, variable-length IP packets are segmented into fixed-length cells at the inputs. The delay a cell suffered before it is reassembled into a packet and delivered to its destination includes the cell delay discussed in the last subsection and the waiting time at the output reassembly buffer. The average packet delay performance under uniform i.i.d. Bernoulli arrivals for the DRRM, iSLIP and EDRRM switches is simulated [15]. Simulation results show that the average packet delay of EDRRM is always comparable with or smaller than that of DRRM when the switch size is larger than 4, and the average packet delays of DRRM and iSLIP are similar to each other. For an EDRRM switch, packet delay increased with packet length under light load, while under heavy load the average packet delays for packets with different sizes are similar. On the other hand, in a DRRM switch the average packet delay increases linearly with the packet size.

## III. Performance Analysis

In this section we will analyze the delay performance of an EDRRM switch under uniform traffic by using an exhaustive random polling system model.

Since an EDRRM switch is symmetric under uniform arrivals and all the inputs will have the same performance, we will consider one input, say input 0, without loss of generality. After one VOQ is served and becomes empty, the input pointer will keep moving in a fixed order until a free output grants the request from this input followed by the transfer of all the cells in the corresponding VOQ. This is similar to an exhaustive service polling system with $N$ stations. After all the cells in one station (VOQ) are served, the server switches to another station and starts a new service. Since the pointer will not stay at a VOQ if the request is not granted, the service order of the VOQs is not fixed, which we will approximate by a random polling system [18], where the next station polled is determined according to some random criterion.

We say that an input (or output) is *busy* at the beginning of a time slot if in the last time slot this input (or output) is matched with an output (or input) and the corresponding VOQ is not empty by the end of last time slot. Otherwise we say that an input (or output) is *free* at the beginning of a time slot.

To simplify the model, we consider the system as a fully symmetric random polling system and the arrival process to each VOQ is independent and identically distributed. We assume that all station VOQs have the same probability of selection for service after a VOQ is served. This is not in general true because the criterion to determine the next VOQ polled is not memoryless. The input arbiter will check the VOQs in a fixed order to send out a request beginning from the last served VOQ, and the requested output will check inputs for a grant in a fixed order beginning from the last served input. However, an examination of simulation runs indicated that this is a reasonable assumption.

The time for the server to transfer all the cells in a VOQ is the *service period*. After a VOQ is served, the server will switch over to another VOQ and start service. The time taken for the server to switch from one VOQ after service completion to another VOQ for a new service period is the *switch over time*. Specifically, suppose one VOQ of input 0 is served and becomes empty by the end of time slot $t-1$, then in time slot $t$ input 0 begins to search for a new input output matching. In time slot $t+n$ an output gives a grant and the new service starts. Then the switch over period is from time slot $t$ to $t+n$, and the switch over time is $n$.

### A. Average switch over time

During a switch over period, the input arbiter pointer moves to a nonempty VOQ and sends a request to the corresponding output. If the output is free, and the input is the first one in a fixed order among all the inputs sending requests to this output, the request will be granted and the switch over period ends. Otherwise, the pointer will move to the next VOQ and repeat this process.

We make the following assumptions:

1) *Pointer Randomization Assumption:* each input has an equal chance of being pointed by an output pointer, and each VOQ has an equal chance of being pointed by an input pointer; and
2) *Memoryless Assumption:* each output has the same probability of being free (with one exception).

The exception to the second assumption is as follows.

Suppose VOQ $k$ of input 0 was just served and became empty by the end of time slot $t-1$. In time slot $t$, suppose only one output is free, then this output must be output $k$ that has just been released by input 0. Since, under the heavy load traffic assumption, we assume at least one VOQ of input 0 is nonempty, input 0 sends a request for the next busy VOQ to the corresponding output. Since this output is busy and cannot grant the request, input 0 will send a request for the next busy VOQ in time slot $t+1$. If no other output is released and output $k$ is always the only free one, the same thing will happen in each time slot until the input pointer returns to VOQ $k$. In this time slot two alternatives can happen. If VOQ $k$ has new arrivals after its last service, the input pointer stays at VOQ $k$ and VOQ $k$ gets service again. Or if VOQ $k$ is still empty, the pointer skips it and directly moves to the next busy VOQ. Then a new cycle begins. We call the period during which the output just been released is the only free output an *inefficient period*. It will terminate when at least one other output becomes free so that other VOQs of input 0 have a chance to get service.

In contrast to the inefficient period, we name the period from the first time slot in which more than one output is free to the time slot just before input 0 gets a new service as an *efficient period*. During this period there is a higher probability of forming a stable, longer lasting matching.

During an inefficient period, the same VOQ can get service several times in succession if it has new arrivals. But these services are typically very short compared to both the service period after an efficient period and the time without service during an inefficient period. The average number of cells available for one service in an inefficient period is at most the product of the arrival rate of one VOQ and $N-1$. For uniform traffic, this value is always less than one. To simplify the analysis, we can omit the services during inefficient periods and consider them part of a switch over time. A switch over period can be an efficient period (if more than one output is free at time slot $t$) or an inefficient period followed by an efficient period (if only one output is free at time slot $t$).

We define $X$ and $Y$ as the length of an efficient period and the length of an inefficient period, respectively, and $m$ as the number of free outputs in a time slot. Then the switch over time

$$S = \begin{cases} X, & \text{if } m > 1 \text{ in time slot } t \\ X+Y, & \text{if } m = 1 \text{ in time slot } t, \end{cases} \quad (1)$$

and

$$E(S) = E(X) + E(Y)P(m = 1 \text{ in time slot } t). \quad (2)$$

We define $\rho$ as the probability that an input slot has an arrival. For symmetric stable traffic, $\rho$ is also the probability that one output or input is busy in a time slot. In time slot $t$, $m = 1$ means that all other inputs are busy, so that $P(m = 1 \text{ in time slot } t) = \rho^{N-1}$.

We first consider $X$. Suppose $X$ begins at $t$ and the next new service starts at $t+n$, then

$$P(X = n) = \begin{cases} q, & n = 0 \\ (1-q)(1-Q)^{n-1}Q, & n > 0, \end{cases} \quad (3)$$

where $q$ is the probability that input 0 gets a grant in time slot $t$, and $Q$ is the probability that input 0 gets a grant in time slot $t+j$, $j > 0$. Note that $m > 1$ in time slot $t$ and $m > 0$ in time slot $t+j$. $Q$ can be expressed as

$$\begin{aligned} Q &= \sum_{m=1}^{N} P(\text{input 0 gets a grant and } \\ &\quad m \text{ inputs are free in this time slot}) \\ &= \sum_{m=1}^{N} P(\text{input 0 gets a grant}/m \text{ inputs are free}) \\ &\quad P(m \text{ inputs are free}) \end{aligned} \quad (4)$$

We already know that input 0 is free, so that

$$P(\text{m inputs are free}) = \binom{N-1}{m-1} \rho^{N-m}(1-\rho)^{m-1}. \quad (5)$$

The fact that $m$ inputs are free in a time slot means that $m-1$ other inputs along with input 0 are sending requests, while $m$ outputs are free. The probability that the output requested by input 0 is free is $\frac{m}{N}$. If there are $i$ other inputs also sending requests to the same output as input 0, the chance that input 0 wins is $\frac{1}{i+1}$. The probability that $i$ other inputs request the same output as input 0 is $\binom{m-1}{i}(1-w)^{m-i-1}w^i$, where $w$ is the probability that an input requests the same output as input 0. Therefore,

$$\begin{aligned} &P(\text{input 0 gets a grant}/m \text{ inputs are free}) \\ &= \frac{m}{N}\sum_{i=0}^{m-1}\binom{m-1}{i}(1-w)^{m-i-1}w^i\frac{1}{i+1} \end{aligned}$$

If an input requests the same output as input 0 does, the corresponding VOQ must be nonempty. A lower bound on the probability that a VOQ is nonempty is $\frac{\rho}{N}$. If there are $k$ other nonempty VOQs, then the probability that this VOQ is selected by the input arbiter is $\frac{1}{k+1}\cdot\binom{N-1}{k}(1-\frac{\rho}{N})^{N-k-1}(\frac{\rho}{N})^k$ is the probability that $k$ of the other $N-1$ VOQs are nonempty. Therefore,

$$\begin{aligned} w &= \frac{\rho}{N}\sum_{k=0}^{N-1}\binom{N-1}{k}\left(1-\frac{\rho}{N}\right)^{N-k-1}\left(\frac{\rho}{N}\right)^k\frac{1}{k+1} \\ &= \frac{1}{N}\left[1-(1-\frac{\rho}{N})^{N-1}\right] \end{aligned} \quad (6)$$

Then

$$\begin{aligned} Q &= \sum_{m=1}^{N}\binom{N-1}{m-1}\rho^{N-m}(1-\rho)^{m-1}\frac{m}{N} \\ &\quad \sum_{i=0}^{m-1}\binom{m-1}{i}(1-w)^{m-i-1}w^i\frac{1}{i+1} \\ &= \frac{1}{Nw}\left[1-(1-w)(1-w(1-\rho))^{N-1}\right] \end{aligned} \quad (7)$$

Similarly, in time slot $t$ input 0 and $m-1$ other inputs request new matches. The output just released by input 0 is free and input 0 requests another output in time slot $t$. Therefore,

$$\begin{aligned} q &= \sum_{m=2}^{N} P(\text{input 0 gets a grant and} \\ &\quad m-1 \text{ other inputs are free in time slot t} \\ &\quad /\text{at least one other input is free}) \\ &= \sum_{m=2}^{N} P(\text{input 0 gets a grant} \\ &\quad /m \text{ inputs are free}) P(m \text{ inputs are free}) \\ &\quad /P(\text{at least one other input is free}) \\ &= \frac{1}{1-\rho^{N-1}} \sum_{m=2}^{N} \binom{N-1}{m-1} \rho^{N-m}(1-\rho)^{m-1} \\ &\quad \frac{m-1}{N-1} \sum_{i=0}^{m-1} \binom{m-1}{i} (1-w)^{m-i-1} w^i \frac{1}{i+1} \\ &= \frac{1}{w(1-\rho^{N-1})(N-1)} [1-(1-w) \\ &\quad (1-w(1-\rho))^{N-1} + \frac{(1-w(1-\rho))^N - 1}{N(1-\rho).}] \end{aligned} \quad (8)$$

From (3), we get

$$E(X) = \sum_{n=1}^{\infty} n(1-q)(1-Q)^{n-1}Q = \frac{1-q}{Q}. \quad (9)$$

We next consider $Y$. Suppose $Y$ begins at $t$,and during time period $[t, t+n-1]$ only output $k$ is free and at time $t+n$ more than one outputs are free. Then

$$P(Y=n) = \rho^{n(N-1)}(1-\rho^{N-1}), \quad \text{and} \quad (10)$$

$$E(Y) = \sum_{n=1}^{\infty} n\rho^{n(N-1)}(1-\rho^{N-1}) = \frac{\rho^{N-1}}{1-\rho^{N-1}}. \quad (11)$$

From (2), (9) and (11), we get

$$E(S) = \frac{1-q}{Q} + \frac{\rho^{2(N-1)}}{1-\rho^{N-1}}. \quad (12)$$

For $E(S^2)$, we will not present the details of derivation, but only the final expression,

$$E(X^2) = \frac{1-q}{Q}\left[\frac{2(1-Q)}{Q} + 1\right], \quad (13)$$

$$E(Y^2) = \frac{\rho^{N-1}}{1-\rho^{N-1}}\left(\frac{2\rho^{N-1}}{1-\rho^{N-1}} + 1\right), \quad (14)$$

$$E(S^2) = E(X^2) + \rho^{N-1}(E(Y^2) + 2E(X)E(Y)) \quad (15)$$

### B. Average delay

In [18] the delay of a random polling system is analyzed. For a fully symmetric system, using the notation in [18], the average delay for a cell is described as

$$E(T) = \frac{1}{2}\left[\frac{\delta^2}{r} + \frac{\sigma^2}{(1-N\mu)\mu} + \frac{Nr(1-\mu)}{1-N\mu} + \frac{(N-1)r}{1-N\mu}\right] \quad (16)$$

where $\mu$ is the arrival rate for one VOQ, $\sigma^2$ is the variation of the arrival process for one VOQ, and $r = E(S)$, $\delta^2 = Var(S) = E(S^2) - E^2(S)$. For each VOQ, under i.i.d. Bernoulli traffic, $\mu = \frac{\rho}{N}$, $\sigma^2 = \frac{\rho}{N}$.

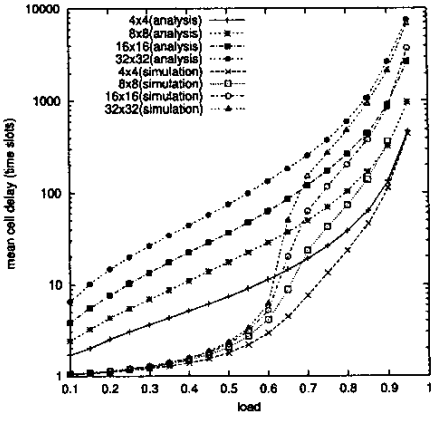


Fig. 2. The average delay of EDRRM with different switch sizes

Our system is not exactly the same as a random polling system in [18]. In a random polling system, after one station is served, the server may switch to an empty station which leads to a service period of zero length, following which a new switch over period begins. However, in our system only nonempty VOQs are considered. An input only requests service for a nonempty VOQ, so that the length of a service period is always larger than zero. Therefore, when using the less efficient random polling system we expect the delay to be over estimated for light traffic load. The analysis is more accurate in predicting the performance under heavy traffic load when VOQ's are less likely to be empty. Also, as the switch size increases, the analysis will approximate the system better since the pointer randomization assumption and the memoryless assumption is closer to reality.

Figure 2 is the comparison of the analysis result of the average delay $E(T)$ and simulation result. The analysis result is quite close to the simulation result when the load is heavy, and is larger than the simulation result when the load is light. The reason for the difference under light load is described above.

### C. The Performance When N Is Large

When $N$ goes to infinity, we will show that both the average switch over time and its second moment converge to a limit. We will also show that for large $N$ the average delay is a function of $N$ and $\rho$ which always has a finite value for all $\rho < 1$.

Since $w$ goes to $\frac{1}{N}(1-e^{-\rho})$ and $(1-w(1-\rho))^N$ goes to $e^{-(1-\rho)(1-e^{-\rho})}$ when $N$ is large, from (8) and (9) we get

$$\lim_{N\to\infty} Q = \lim_{N\to\infty} q = \frac{1-e^{-(1-\rho)(1-e^{-\rho})}}{1-e^{-\rho}}. \quad (17)$$

Therefore,

$$\lim_{N\to\infty} E(S) = \frac{1-e^{-\rho}}{1-e^{-(1-\rho)(1-e^{-\rho})}} - 1. \quad (18)$$

Also,

$$\lim_{N\to\infty} E(S^2) = \lim_{N\to\infty} (2E^2(S) + E(S)). \quad (19)$$

Similarly, for large $N$, it can be shown that

$$E(T) \to E(S)\frac{N-\rho}{1-\rho} + \frac{2-\rho}{2(1-\rho)} \sim E(S)\frac{N}{1-\rho}. \quad (20)$$

Since $E(S)$ has a finite limit, $E(S^2)$ also has a finite limit for $\rho < 1$. Similarly, $E(T)$ is linear in $N$ when $N$ is large, and finite for $\rho < 1$. These results support our conjecture that the switch throughput approaches 100% for large $N$ under uniform traffic.

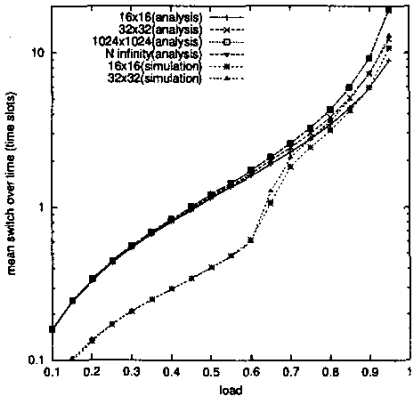


Fig. 3. The average switch over time of EDRRM with different switch sizes

Figure 3 shows the average switch over time for 4 different switch sizes and its limit, compared to simulation results. It can be seen that when the switch size is 1024, the average switch over time is almost identical to the limit.

Figure 4sh ows the calculated average delay $E(T)$ of 4 switches of large size.

## IV. Conclusions

The EDRRM algorithm is a variation of the DRRM scheduling algorithm. The implementation complexity of EDRRM's switching fabric is the same as that of DRRM, while packet reassembly is simpler than most other popular matching schemes. In an EDRRM switch when an input is matched with an output all the cells in the corresponding VOQ are served continuously before any other VOQ of the same input can be served. The average cell delay of an EDRRM switch is analyzed by using an exhaustive random polling system model in this paper. The performance of an EDRRM switch is comparable to or better than a DRRM switch or an iSLIP switch for most traffic scenarios. Under uniform i.i.d. traffic, an EDRRM switch has a

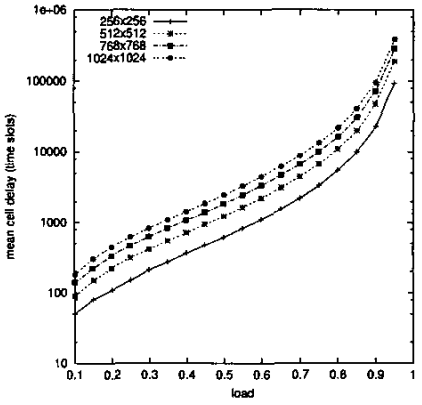


Fig. 4. The average delay for large switch sizes

larger average cell delay than a DRRM switch, but its average packet delay is lower and not sensitive to either switch size or packet size [14]. Furthermore, in [14] we showed that EDRRM is not sensitive to traffic burstiness. Under nonuniform traffic the throughputs of a DRRM switch and an iSLIP switch drop well below 100%, while the throughput of an EDRRM switch is closer to 100%.

## References


[1] M. J. Karol, M. Hluchyj, and S. Morgan, "Input vs. output queuing on a space-division packet switch", *Proc. GLOBECOM* 1986, pp. 659-665.

[2] M. J. Karol, M. Hluchyj, and S. Morgan, "Input versus output queuing on a space-division packet switch," *IEEE Trans. on Communications*, vol.35, pp. 1347-1356, 1987.

[3] L. Tassiulas, A. Ephremides, "Stability properties of constrained queueing systems and scheduling for maximum throughput in multihop radio networks," *IEEE Trans. Automatic Control*,Vol. 37, No. 2, pp. 1936-1949.

[4] N. McKeown, V.A nantharam, and J. Walrand, "Achieving 100% throughput in an input-queued switch," *IEEE INFOCOM'96*,p p. 296-302.

[5] N. McKeown, A. Mekkittikul, V. Anantharam and J. Walrand, "Achieving 100% throughput in an Input-Queued switch", *IEEE Trans. Communications*, vol. 47, No. 8, pp. 1260-1267, Aug. 1999.

[6] N. McKeown, "Scheduling algorithms for input-queued cell switches", *Ph.D. Thesis*,U C Berkeley, May 1995.

[7] A. Charny, P. Krishna, N. Patel and R. Simcoe, "Algorithms for providing bandwidth and delay guarantees in Input-Buffered crossbars with speedup", *IWQOS'98*, May 1998.

[8] P. Krishna, N. S. Patel, A.Charny and R. Simcoe, "On the speedup required for work-conserving crossbar switches", *IWQOS'98*, May 1998.

[9] A. Mekkittikul and N. McKeown, "A practical scheduling algorithm to achieve 100% throughput in input-queued switches", *IEEE INFOCOM 98*, Vol 2, pp. 792-799, April 1998.

[10] T.E. Anderson, S. S. Owicki, J. B. Saxe and C. P.Th acker, "High speed switch scheduling for local area networks," *ACM Trans. on Computer Systems*, vol. 11, No. 4, pp. 319-352, Nov.1 993.

[11] N. McKeown, "The iSLIP scheduling algorithm for Input-Queued switches", *IEEE/ACM Trans. Networking*, vol. 7, pp. 188-201, April 1999.

[12] H. J. Chao, "Saturn: a terabit packet switch using Dual Round-Robin", *IEEE Communication Magazine*, vol. 38 12, pp. 78-84, Dec. 2000.

[13] Y. Li, S. Panwar, H. J. Chao, "On the performance of a Dual Round-Robin switch," *IEEE INFOCOM 2001*, vol. 3, pp. 1688-1697, April 2001.

[14] Y. Li, S. Panwar, H. J. Chao, " The Dual Round-Robin Matching switch with exhaustive service," *2002 Workshop on High Performance Switching and Routing (HPSR 2002)*,Ma y 2002.

[15] M. A. Marsan, A. Bianco, P. Giaccone, E. Leonardi, F. Neri, "Packet Scheduling in Input-Queued Cell-Based Switches," *IEEE INFOCOM 2001*, vol. 2, pp. 1085-1094, April 2001.

[16] H. Takagi, "Queueing analysis of polling models: an update," *Stochastic Analysis of Computer and Communication Systems*, New York: Elsevier Science and B. V. North Holland, pp. 267-318 1990.

[17] C-S. Chang, D. Lee and Y. Jou, "Load balanced Birkhoff-von Neumann switches, part I: one-stage buffering ," *Special issue of Computer Communications on "Current Issues in Terabit Switching,"* 2001.

[18] L. Kleinrock, H. Levy, "The analysis of random polling systems," *Operations Research*, Vol.36, No.5 (September-October), pp. 716-732 1988.