고성능 입력 큐스위치를 위한 통제된 슈도요구 이중화 라운드로빈 매칭 조정 알고리즘

준회원 남 미 화*, 정회원 김 덕 년**

Well-Regulated Pseudo-request Dual Round-Robin Matching Arbitration Algorithm for High Performance Input-Queued Switches

MeiHua Nan*, Doug Nyun Kim** *Regular Members*

요 약

고속 스케줄링 기법은 입력 큐 스위치의 성능을 극대화하기 위해 필요하다. 가상 출력큐 스위치 구조에 관하여 지금까지 iSLIP, DRRM과 같은 라운드로빈 스케줄링 기법이 제안되어 왔다. iSLIP방식은 높은 성능을 보여주고 있고 이미 하드웨어 구현이 되어 있고 DRRM 또한 iSLIP보다 간단하고 성능이 더 좋음을 나타내고 있지만, 둘다 라운드로빈의 포인터동기 문제를 효과적으로 풀지 못하고 있다. 본 논문에서 제안된 알고리즘은 DRRM 알고리즘에 기초하여 새롭게 제안되었고 포인터를 항상 비동기되도록 유지하는 특성이 있다. 또한 라운드로빈 방식을 그대로 견지하므로써 구현이 간편한 특성을 가지고 있다. 시뮬레이션 결과에 의하면 제안된 알고리즘은 다양한 트래픽 모델에서 iSLIP이나 DRRM보다 더 좋은 성능을 보여주고 있다.

ABSTRACT

High-speed scheduling algorithms are required for high-performance input-queued switches to achieve good performance. Various Round-Robin scheduling algorithms for Virtual-Output-Queue (VOQ) switch architectures have been proposed, like iSLIP, DRRM (Dual Round-Robin Matching). iSLIP can achieve high performance and have already been implemented in hardware. DRRM has been proved to achieve better performance and simpler than iSLIP. But neither iSLIP nor DRRM can efficiently solve the problem of the Round-Robin pointers' synchronization. In this paper, we have proposed "Well-Regulated Pseudo-request Dual Round-Robin Matching" Algorithm. It is developed from DRRM, and can always keep the pointers' desynchronization. Since our algorithm is based on the Round-Robin scheduling, it is also simple to be implemented. And simulation results also show that our proposed algorithm performs pretty well under various traffic models.

Key Words : Well-regulated, Pseudo-request, VOQ, iSLIP, DRRM.

I. Introduction

In the recent past, many research efforts were devoted to design the efficient ATM switches for B-ISDN [1]. Various cell switch architectures have been proposed in the literature, and most of them can be implemented [2,3]. For high-performance systems, the desirable properties of schedulers are:

- *High Throughput:* schedulers should serve as many queues as possible; no output port may be idle as long as it can serve some

* 명지대학교 통신공학과
논문번호 : KICS2004-06-062, 접수일자 : 2004년 6월 27일

input cell destined to it.

- *Fairness:* no input queue should remain unserved indefinitely, i.e. starvation-free.
- *Fast:* schedulers should be fast and not become the performance bottleneck; the scheduler should therefore find a match as fast as possible.
- *Implementation simplicitz:* schedulers should be amenable to simple hardware implementation, preferably within a single chip.

The traditional output-queued (OQ) switches have the best possible delay-throughput performance for all traffic distributions, however, an N-time speedup fabric is required. In contrast, input-queued (IQ) switches don't require any speed up, and relax the memory-bandwidth constraints. So it is cost-effective for high-speed switches. Unfortunately, when using a first-in-first-out (FIFO) queueing discipline at the input queues, due to the head-of-the-line (HoL) blocking problem, the maximum throughput is limited to 58.6% [4] under uniform traffic and much lower than that for other traffic models.

Virtual-Output-Queue (VOQ) architecture [2] is proposed to solve the HoL problem while achieving the scalability of IQ switches. Rather than maintaining a single FIFO queue for all cells, each input maintains a separate queue for the cells destined to different outputs. An input-queued switch with VOQ is shown in Fig. 1. In this architecture, the switch performance essentially depends on the scheduling algorithm, that is, the arbitration between the input ports and the output ports. A good algorithm should achieve high performance, which we have summarized above (4 items).

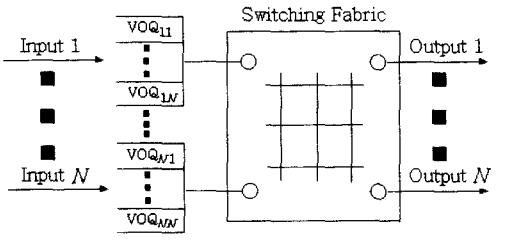


Fig. 1. An input-queued switch with VOQ

Many scheduling algorithms have been proposed in the literature. They can be classified into two types: approximating maximum size matching (MSM) and approximating maximum weight matching (MWM). Although the MWM algorithms have been shown that they can achieve 100% throughput under any traffic [5,6], they are too complex to be implemented in hardware and have a time complexity of $O(N^3 \log^N)$. So in this paper we concentrate on some more practical MSM algorithms, including iSLIP [7], DRRM [8,9] and etc. Round-Robin Pointers' desynchronization plays a very important role in these algorithms. Trying to get the best effort of simplicity, desynchronization and maximum matching, we propose a Well-Regulated Pseudo-request Dual Round-Robin Matching algorithm, which performs pretty well under various traffic models. "Well-Regulated" means Round-Robin pointers' initialization and moving are very regular. The explanations of pointers' desynchronization, initialization and moving are shown in the following sections in detail.

The rest of the paper is organized as follows: Section Ⅱ introduces the typical scheduling algorithms that approximate MSM. Section Ⅲ describes our proposed algorithm. The performance is shown in Section Ⅳ. Finally, Section Ⅴ concludes the paper.

Ⅱ. Related Typical Scheduling Algorithms

A group of scheduling algorithms has been proposed to approximate MSM, and they can be simply implemented. Here we introduce two typical ones: iSLIP and DRRM. We must know iSLIP and DRRM are both based on Round-Robin scheduling. Round-Robin scheduling means that each Round-Robin has a pointer which points to the current element (starting point for current time slot), and the pointer can move around the Round-Robin in the next time slot according to some scheduling schemes.

1. iSLIP Algorithm

Well-known iSLIP algorithm was proposed based on RRM (Round-Robin Matching) [10]. RRM works with three phases: input request, output grant (using grant Round-Robin) and input accept (using accept Round-Robin). It has already been shown that RRM algorithm can't perform well because of the grant Round-Robin pointers' synchronization and its highest throughput under uniform traffic is just bound to 63%. iSLIP only does a small change to the grant phase of RRM, but much better performance is achieved. The detailed description for iSLIP is as follows:

Step 1: Request. Each input sends a request to every output for which it has a queued cell (the relative VOQ is not empty).

Step 2: Grant. If an output receives any requests, it chooses the one that appears next in a fixed, Round-Robin schedule, starting from the current position of the pointer (the current position of the pointer also can be called as starting point). The output notifies each input whether or not its request was granted. The pointer g_i is incremented (modulo N) to one location beyond the granted input if, and only if, the grant is accepted in Step 3.

Step 3: Accept. If an input receives any grants, it accepts the one that appears next in a fixed, Round-Robin schedule, starting from the current position of the pointer (starting point). The pointer a_i is incremented (modulo N) to one location beyond the accepted output.

In the grant phase, the pointer g_i of RRM is always incremented (modulo N) to one location beyond the granted input regardless of whether the grant is accepted or not. So it increases the output grant pointers' synchronization and leads to the performance degradation. In contrast, iSLIP algorithm reduces the synchronization and achieves high performance. It can get 100% throughput under uniformly distributed i.i.d. traffic. Furthermore, iSLIP algorithm provides fairness and prevents starvation. An example for iSLIP is shown in Fig. 2.

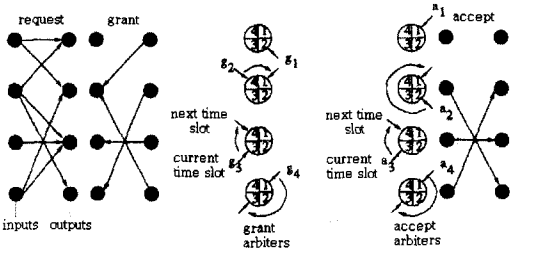


Fig. 2. An example for iSLIP algorithm

2. Dual Round-Robin Matching Algorithm (DRRM)

DRRM algorithm uses request Round-Robin for input arbiters and grant Round-Robin for output arbiters. So it only needs two steps to setup exclusive connection between the input port and relative output port. A detailed description of this two-step algorithm is as follows:

Step 1: Request. Each input sends an output request corresponding to the first nonempty VOQ in a fixed round-robin order, starting from the current position of the pointer (starting point). The pointer remains at that nonempty VOQ if the selected output is not granted in step 2. The pointer of the input arbiter is incremented (modulo N) to one location beyond the selected output if, and only if, the request is granted in step 2.

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 (starting point). The output notifies each requesting input whether or not its request was granted. The pointer of the output arbiter is incremented (modulo N) to one location beyond the granted input. If there are no requests, the pointer remains where it is.

An example for DRRM is shown in Fig. 3. DRRM achieves a saturation throughput of 100% with lower delay and provides better fairness than iSLIP. It also has lower implementation complexity compared to other algorithms with similar performance.

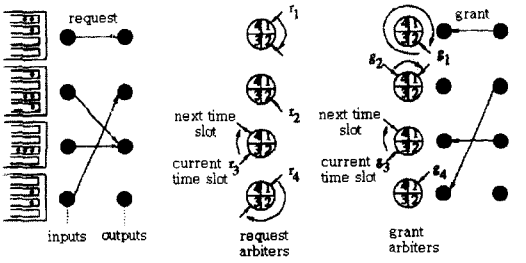


Fig. 3. An example for DRRM algorithm

3. Pointers' Desynchronization

We can see that the main point for the performance of these algorithms lies in how to update the pointers (grant, accept pointers for RMM and iSLIP; request pointer for DRRM). Pointers' desynchronization as fast as possible is desired. Although iSLIP and DRRM perform well under uniformly distributed i.i.d traffic, they haven't efficiently solved the problem of pointers' synchronization.

Ⅲ. Our Proposed Algorithm

Why does the pointers' synchronization degrade the performance and how do we solve the pointers' synchronization problem described above? Since DRRM is better than iSLIP, our work is developed based on DRRM.

To study the synchronization effect, We make the definitions as follows:

a). p_r: two input arbiters select the requests directed to same output;

b). r_i: starting point of input i's request round-robin,
r_j: starting point of input j's request round-robin,
$r_i \leqq r_j (r_i = r_j - d \; for 0 \leqq d < N)$;

c). Suppose a VOQ is empty with propability δ and not empty with $1-\delta$;

d). Q_{io}: input i selects the request for output o, there exists three cases:
1) $1 \leqq o < r_i$, 2) $r_i \leqq o < r_j$,
3) $r_j \leqq o \leqq N$;

In case 1), input arbiter select Q_{io} if $Q_{ir_i}, Q_{i(r_i+1)}, \cdots, Q_{iN}, Q_{i1}, Q_{i2}, \cdots, Q_{i(o-1)}$ are empty and Q_{io} is not empty. Therefore, input arbiter selects Q_{io} with probability $\delta^{N-r_i+o}(1-\delta)$. Since the arbitrations of both input i and j are independent of each other, for case 1), two input arbiter i and j select cells directed for same output o with probability $\sum_{m=1}^{r_i-1} \delta^{N-r_i+m}\delta^{N-r_j+m}(1-\delta)^2$.

Similarly for case 2), two input arbiters i and j select cells directed to same output o with probability $\sum_{m=r_i}^{r_j-1} \delta^{m-r_i}\delta^{N-r_j+m}(1-\delta)^2$;

for case 3), two input arbiters i and j select cells directed to same output o with probabilities $\sum_{m=r_j}^{N} \delta^{m-r_i}\delta^{m-r_j}(1-\delta)^2$.

So p_r can be expressed like this:

$$p_r = (1-\delta)^2 \Big(\sum_{m=1}^{r_i-1} \delta^{N-r_i+m}\delta^{N-r_j+m} + \sum_{m=r_i}^{r_j-1} \delta^{m-r_i}\delta^{N-r_j+m} + \sum_{m=r_j}^{N} \delta^{m-r_i}\delta^{m-r_j} \Big). \quad (1)$$

Since $r_i = r_j - d$, then p_r can be modified as:

$$p_r = \frac{(1-\delta)(1-\delta^N)}{1+\delta} \times (\delta^d + \delta^{N-d}) \quad (2)$$

Now we want to get the condition which makes p_r maximized. When N is fixed, $\frac{(1-\delta)(1-\delta^N)}{1+\delta}$ is a constant and the value is always $\geqq 0$ for any δ where $0 < \delta \leqq 1$. So we only consider $(\delta^d + \delta^{N-d})$.

Let $p_0 = \delta^0 + \delta^{N-0} = \delta^0 + \delta^N \; (d=0)$,
$p_d = \delta^d + \delta^{N-d} \; (0 < d < N)$

$$p_0 - p_d = (1-\delta^d) - \delta^{N-d}(1-\delta^d)$$
$$= (1-\delta^d)(1-\delta^{N-d}) \geq 0 \; for (0 < d < N) \quad (3)$$

So $p_0 \geq p_d for 0 < d < N.$ (4)

Combining the equation (2) and (4), it is obvious that p_r is maximized at $d = 0$ for any δ where $0 < \delta \leq 1$. This means if the starting points of two individual input request arbiters are same, then the probability that selected requests of the two arbiters directed to a same output is maximized. That is, the performance will be degraded.

For the reason we described above, we propose a new input request arbitration to minimize p_r : the starting points of input request arbiters are different from each other at any time slot. So we can set the initial starting point of each input arbiter like this:

$$r_i = i, \quad i = 1, 2, \ldots, N.$$

The same order will also be set to the output initial grant points. That is,

$$g_i = i, \quad i = 1, 2, \ldots, N.$$

After that, at the following each time slot, the request pointer of each individual input arbiter is always incremented by one (modulo *N*) in a clockwise rotation; the grant pointer's rotation is the same as the rotation of request pointer.

We know that the proposed scheme can fully solve the synchronization problem under fully-loaded (none of the VOQs is empty) traffic, but in other cases, more than one request directed to a same output also exists. So in order to improve the number of matching in a time slot, we considered a novel mechanism and added it to our scheme. We call it "Pseudo-request".

Up to now, we have described our main idea. So the proposed algorithm can be expressed in detail as following steps:

Initialization:

Set initial input request arbiter round-robin pointer as: $r_i = i, \quad i = 1, 2, \ldots, N.$

Set initial output grant arbiter round-robin pointer as: $g_i = i, \quad i = 1, 2, \ldots, N.$

Execution:

Step 1: Request.

Each input sends an output request corresponding to the first nonempty VOQ in a fixed round-robin order, starting from the current position of the request pointer.

Each input also has a Pseudo-request pointer, which points to the request VOQ that was rejected in the previous time slot. Each input also sends a Pseudo-request to the output the Pseudo-request pointer points to.

If the input synchronously receives a grant and Pseudo-grant in Step 2, it ignores Pseudo-grant; otherwise it will accept the Pseudo-grant. The request pointer of the input arbiter will be incremented by one (modulo *N*) in clockwise rotation whether or not the request is granted. The Pseudo-request pointer will be inactivated if it has kept unmoved for a certain time slots. Until a new request rejection, the Pseudo-request pointer will be activated.

Step 2: Grant.

If an output synchronously receives requests and Pseudo-requests, it ignores the Pseudo-requests and chooses the one that appears next in a fixed round-robin schedule, starting from the current position of the grant pointer. The output notifies each requesting input whether or not its request was granted. Otherwise, it will randomly choose one Pseudo-request and send Pseudo-grant.

The grant pointer of the output arbiter will be incremented by one (modulo *N*) in clockwise rotation.

Fig. 4 shows an example of our proposed Well-Regulated Pseudo-request Dual Round-Robin Matching arbitration algorithm.

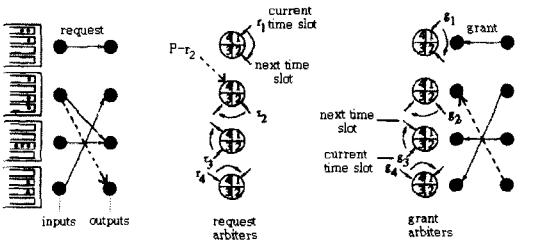


Fig. 4. An example for our proposed algorithm

Ⅳ. The Performance Analysis

This section focuses on the fairness, delay and throughput performance of our proposed algorithm.

1. Fairness

Our algorithm is fair and doesn't suffer from starvation. Because our algorithm is developed from DRRM (Round-Robin based algorithm) and in order to provide service fairness and maximum matching at the end of each time slot, a pseudo-request mechanism is adopted and both the starting points of request arbiter and grant arbiter are increased (modulo N) by one in clockwise rotation.

2. Delay

We have done the simulations under various traffic models. In order to get a good estimate of the performance measure, simulations have been run for sufficiently long time (5×10^5 cell times). Speedup is not considered in this paper, and iSLIP algorithm is considered only with single iteration for a fair comparison. It is shown that our algorithm performs pretty better than iSLIP and DRRM.

2.1. Case 1: Under Bernoulli Traffic

As in [7], under heavy load, our proposed algorithm serves each VOQ once every N cycles. So the queues approximate an M/D/1 queue with arrival rates $\frac{\lambda}{N}$ and deterministic service of N cell times length. Under heavy load of Bernoulli traffic, the approximate delay D for arrival rate λ per slot time can be approximated by:

$$D = \frac{\lambda N}{2(1-\lambda)} \tag{5}$$

The above expression explains that under heavy traffic loads, delay is roughly linearly increased with N. The main cell delay for M/D/1 system compared to our proposed switch is shown in Fig. 5 with N = 8 and 16 respectively.

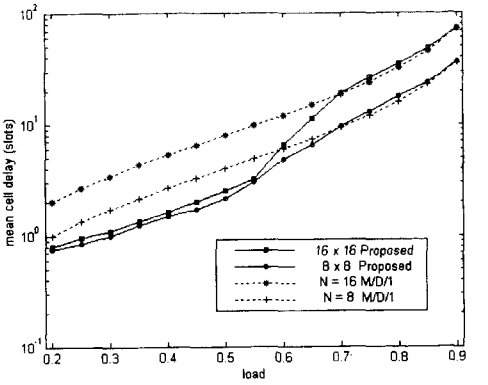


Fig. 5. Comparison of the average delay for our proposed switch with M/D/1 system when N = 8, 16 under uniform i.i.d. Bernoulli traffic.

Fig. 6 shows the performance improvement of our proposed algorithm over iSLIP and DRRM in a 16×16 switch under uniform i.i.d. Bernoulli traffic.

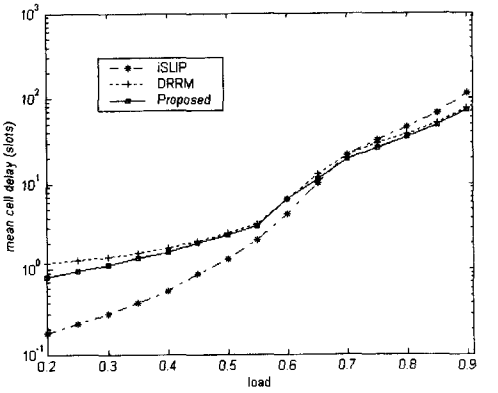


Fig. 6. Comparison on the performance of our proposed algorithm with iSLIP, DRRM when switch size is 16 under i.i.d. uniform Bernoulli traffic.

Although under moderate load, our proposed scheme performs a bit worse than iSLIP (the difference is less than 1 time slot), under heavy load ($\rho > 0.65$) it can be seen that our algorithm performs much better than iSLIP. When ρ is close to 0.9, our proposed algorithm can reduce the delay about 40 time slots. On the whole, our algorithm also always performs better than DRRM.

2.2. Case 2: Under Bursty Traffic

Many ways of modeling bursts in network traffic have been proposed [11,12,13,14]. We illustrate the effect of burstiness using an on-off markov-modulated process.

Fig. 7 compares on the performance of our proposed algorithm and iSLIP, DRRM algorithms in a 16×16 switch with respective average burst length 16 and 32. Note that the delays increase approximately linearly with burst length.

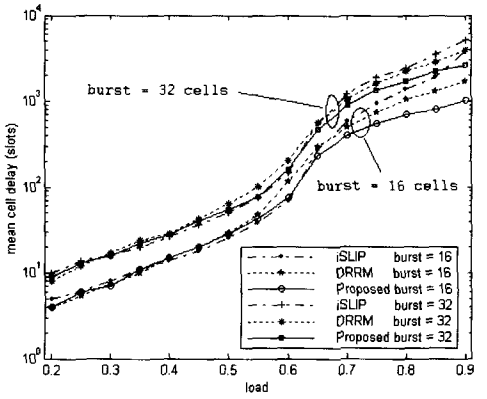


Fig. 7. Comparison on the performance of our proposed algorithm with iSLIP, DRRM when switch size is 16 under uniform bursty traffic of average burst length 16 and 32.

From the figure it can be seen our proposed algorithm performs better than iSLIP and DRRM under these both models, especially when the traffic load is heavy. The reason is that our algorithm from the arbiters' initialization is based on the input requests' desynchronization. Under heavy traffic, it can perform fully desynchronizing from the start, so the burstiness reduction is also quickly implemented.

2.3. Case 3: Under Hot-Spot Traffic

Here we will discuss the delay performance under one typical non-uniform traffic - the so-called "hot-spot" case.

For an example 4×4 switch, the definition is shown as in Fig. 8. If output 4 is the "hot-spot" output and each flow is Bernoulli. It means that the "hot-spot" output has twice as much load as the other outputs.

Input \ Output	1	2	3	4
1	x	x	x	2x
2	x	x	x	2x
3	x	x	x	2x
4	x	x	x	2x

Fig. 8. The definition of "hot-spot" case for 4×4 switch

We use the same definition for a 16×16 switch and get the result as Fig. 9.

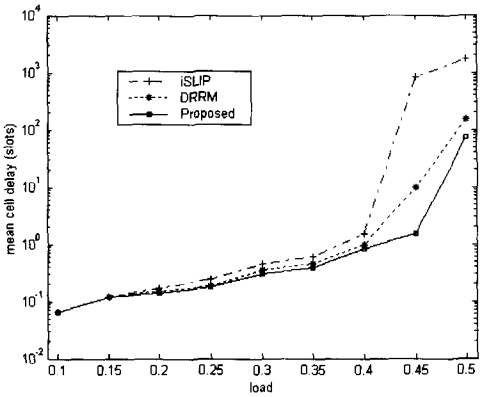


Fig. 9. Comparison on the performance of our proposed algorithm with iSLIP, DRRM under "hot-spot" case.

Because we only consider admissible traffic for this "hot-spot" case simulation, which means that no input or output is overloaded. So the input load is maximized at 0.5. It is obvious that our algorithm performs pretty well and much better than iSLIP, DRRM. The reason is mainly that following the increase of the traffic load, input pointers' desynchronization and maximum matching are kept. So the hot-spot output 16 always gets a chance to be served. The constant service for hot-spot output assures that the cells in the VOQs related to this hot-spot output don't need to wait for a quite long time, thus the delay performance is improved.

3. Throughput

It has already been proved that under uniformly

distributed heavy i.i.d. traffic such that none of the VOQs empty at any time, the maximum throughput of iSLIP and DRRM is 100%. Our algorithm also can get the 100% throughput under such case. The reason is very simple. Our scheme assures the input pointers' desynchronization at any time, so under heavy traffic case, no time is needed to implement the pointers' desynchronization like iSLIP and DRRM. It also means 100% throughput can be gained.

We also want to compare the throughput of these three algorithms under non-uniform "hot-spot" case. The hot-spot model is defined the same as [15,16]: a high rate of traffic is destined to one hot-spot output and all other traffic is distributed to other outputs uniformly. Define h as the fraction of cells destined to the hot-spot output. Then the load rate ρ for the input port can be expressed as:

$$\rho = \rho h + (1-h)\rho \qquad (6)$$

For one input, a cell rate of ρh is destined to the hot-spot output and $(1-h)\rho$ is destined uniformly to the other N-1 outputs. Thus from all the inputs, there is a total cell rate of rhN destined to the hot-spot output.

We have done the simulation for any h in the range $(\frac{1}{N}, 1)$ when the case ρ is 1.0. The result is shown in Fig. 10.

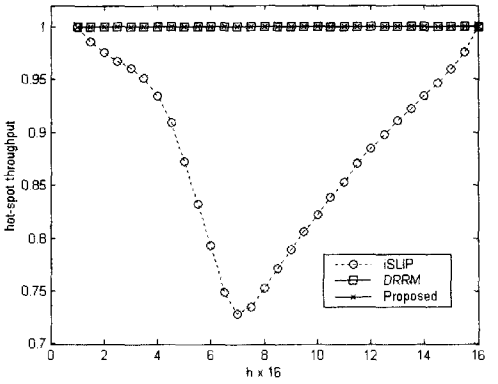


Fig. 10. Comparison on the throughput of our proposed algorithm with iSLIP, DRRM for hot-spot output in a 16×16 switch as a function of h.

The hot-spot output throughputs of DRRM and our algorithm are found to be 100% for any h, but the throughput of iSLIP scheme is waved with h. This is due to the fact that with iSLIP, an input can request for multiple outputs at one time, and the granted input by the hot-spot output maybe doesn't accept the grant. But this is different from the situation with DRRM and our algorithm, in which an input will request for at most one output in each time slot, and if the request is granted, the input will always accept it. When more traffic is destined to the hot-spot output, the VOQs related to the hot-spot output have long queues, and only a few cells wait in the VOQs related to the other outputs. Maybe some of the VOQs corresponding to other outputs are empty. So the hot-spot output will always get at least one request from all inputs in each time slot, and no matter which request is granted, one cell will be sent to this hot-spot output.

It has been proved that 100% throughput can be achieved under i.i.d. uniform heavy traffic such that none of the VOQs is empty for iSLIP. Because pointers can be fully desynchronized after a finite time, each input can send a cell to the corresponding output. When h is close to $\frac{1}{16}$, the traffic for iSLIP is nearly uniform, so the throughput is close to 100%. When h lies in $(\frac{1}{16}, \frac{7}{16})$ ($\frac{7}{16}$ is close to the middle value between $\frac{1}{16}$ and 1), the throughput is decreased from 100% to over 70%. The reason is that following the increase of the traffic destined to the hot-spot output, the arbiter pointers are difficult to be fully desynchronized. The granted input by the hot-spot output maybe doesn't accept this grant while accepts another one due to the receipt of more than one grants, thus the throughput performance is degraded. When h lies in $(\frac{7}{16}, 1)$, following the increase of h, more and more traffic will be destined to the hot-spot

output. Although the arbiter pointers almost can't be fully desynchronized, due to the fact that only a few incoming cells are destined to other outputs and therefore more and more VOQs for other outputs will be empty for most of the time, some inputs will likely only request for the hot-spot output, thus the hot-spot output will always can be served. So the throughput performance will be improved with the increase of h till close to 100%. As h gets to 1, it is obvious that the hot-spot output throughput can get 100%. The reason is that the traffic is only destined to the hot-spot output.

VI. Conclusion

VOQ architecture is very necessary to the input-queued ATM switches. Many Round-Robin algorithms which approximate maximum size matching based on VOQ architecture have been proposed. And it is also proved that traditional Round-Robin scheduling algorithms like iSLIP, DRRM and etc can't efficiently solve the problem of Round-Robin pointers' synchronization. So the performance is much degraded. In order to study why the performance is degraded with Round-Robin pointers' synchronization, some analysis has been done by mathematical method. Based on the result and for the purpose of maximum matching, we have proposed a new algorithm: Well-Regulated Pseudo-request Dual Round-Robin Matching algorithm. This proposed algorithm can efficiently solve the synchronization problem of Round-Robin scheduling, and the implementation is also simple. Through the simulation results, it is also shown that our proposed algorithm performs better than other schemes in performance.

References

[1] Awdeh Ra' ed Y., Mouftah H.T., "Survey of ATM Switch Architectures", *Computer Networks & ISDN Szstems*, Vol. 27, No. 12, pp. 1567-1613, May 1995.

[2] T. Anderson, S. Owicki, J. Saxe, and C. Thacker, "High Speed Switch Scheduling for Local Area Networks", *ACM Transactions on Computer Szstems*, Vol. 11, No. 4, pp. 319-352, November 1993.

[3] N. Mckeown, M. Izzard, A. Mekkittikul, W. Ellersick, and M. Horowitz, "Tiny Tera: A Packet Switch Core", *IEEE Micro*, Vol. 17, pp. 26-33, January 1997.

[4] M. Karol, M. Hluchyj, and S. Morgan, "Input versus output queueing on a space division switch", *IEEE Trans. on Commun.*, Vol. 35, No. 12, pp. 1347-1356, December 1987.

[5] N. Mckeown, A. Mekkittikul, V. Anantharam, and J. Walrand, "Achieving 100% Throughput in an Input-Queued Switch", *IEEE Transactions on Communications*, Vol. 47, pp. 1260-1267, August 1999.

[6] A. Mekkittikul, and N. Mckeown, "A Practical Scheduling Algorithm to Achieve 100% Throughput in Input-Queued Switches", *IEEE INFOCOM 98*, San Francisco, April 1998.

[7] N. Mckeown, "The iSLIP scheduling algorithm for input-queued switches", *IEEE/ACM Trans. Networking*, Vol. 7, No. 2, pp. 188-200, April 1999.

[8] H.J. Chao and J.-S. Park, "Centralized contention resolution schemes for a large-capacity optical ATM switch", *Proc. IEEE ATM Workshop' 97*, Fairfax, VA, May 1998.

[9] H.J. Chao, "Saturn: A terabit packet switch using dual round-robin", *IEEE Commun. Mag.*, Vol. 38, No. 12, pp. 78-84, December 2000.

[10] N. Mckeown, "Scheduling Cells in an Input-Queued Switch", PhD thesis, University of California at Berkeley, May 1995.

[11] D. Anick, D. Mitra, and M. M. Sondhi,

"Stochastic theory of a data-handling system with multiple sources", *Bell Szst. Tech. J.*, Vol. 61, pp. 1871-1894, 1982.

[12] H. Heffes and D. M. Lucantoni, "A Markov modulated characterization of packetized voice and data traffic and related statistical multiplexer performance", *IEEE J. Select. Areas Commun.*, Vol. 4, pp. 856-868, 1988.

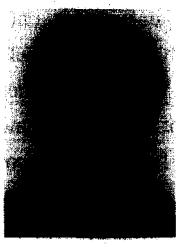
[13] R. Jain and S. A. Routhier, "Packet trains: Measurement and a new model for computer network traffic", *IEEE J. Select. Areas Commun.*, Vol. 4, pp. 986-995, April 1986.

[14] W. E. Leland, W. Willinger, M. Taqqu, D. Wilson, "On the self-similar nature of Ethernet traffic", in *Proc. SIGCOMM*, San Francisco, CA, pp. 183-193, September 1993.

[15] G. F. Pfister, "'Hot Spot' contention and combing in multistage interconnection networks", *IEEE Trans. on Computers*, Vol. C-34, No. 10, pp. 943-948, October 1985.

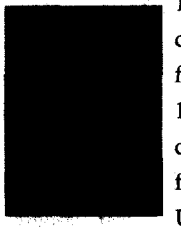
[16] J.S. Park, "Design and Analysis of Large-Capacity Multicast Packet Switches", Ph.D. dissertation, Polytechnic University, January 1998.

MeiHua Nan *Regular Member*



1998"2002: received the B.S. degree in computer science and engineering from Dalian Nationalities University, Dalian, China;
2002"present: pursuing for the M.S. degree in Communication Engineering at Myongji University, Yongin, Korea.
Research interests: ATM Networks, Satellite Communication.

Doug Nyun Kim *Regular Member*



1971"1975: received the B.S. degree in Electrical Engineering from Seoul University, Korea;
1980"1981: received the M.S. degree in Electrical Engineering from SUNY at Stony Brook, USA;
1985"1988: received the Ph.D. degree in Electrical Engineering from Auburn University, USA;
1988"1995: Research fellow with the Electronics and Telecommunications Research Institute (ETRI);
1995"present: Professor with the Department of Communication Engineering, Myongji University.
Research interests: Wireless Communication, High Speed Satellite Communication, Access Protocol.