An efficient single-iteration single-bit request scheduling algorithm for input-queued switches

Bing Hu a, Kwan L. Yeung b, Zhaoyang Zhang a,∗

a Department of Information Science and Electronic Engineering, Zhejiang University, Hangzhou, People's Republic of China
b Department of Electrical and Electronic Engineering, The University of Hong Kong, Hong Kong, People's Republic of China

A R T I C L E   I N F O

Article history:
Received 27 June 2012
Received in revised form 6 September 2012
Accepted 21 September 2012
Available online 7 October 2012

Keywords:
Input-queued switch
Single-iteration scheduling algorithm
Single-bit request scheduling algorithm

A B S T R A C T

Aiming at minimizing communication overhead of iterative scheduling algorithms for input-queued packet switches, an efficient single-iteration single-bit request scheduling algorithm called Highest Rank First with Request Compression 1 (HRF/RC1) is proposed. In HRF/RC1, scheduling priority is given to the preferred input–output pair first, where each input has a distinct preferred output in each time slot. If an input does not have backlogged packets for its preferred output, each of its non-empty VOQs sends a single-bit request to the corresponding output. This single bit distinguishes one longest VOQ from other non-empty VOQs among an input port. If an output receives a request from its preferred input, it grants this input. Otherwise, it gives the higher priority to the longest VOQ than other non-empty VOQs. Similarly, an input accepts the grant following the same propriety sequence. In case of a tie, the winner is selected randomly. Compared with other single-iteration algorithms with comparable communication overhead, we show by simulations that HRF/RC1 always gives the best delay-throughput performance.

© 2012 Elsevier Ltd. All rights reserved.

1. Introduction

Due to the wide use of wavelength division multiplexing technology in fiber, the transmission capacity increases sharply (Liu et al., 2012; Guo, 2007), while the processing capacity of switches increases slowly. The mismatch between the transmission capacity by fiber and the processing capacity by switches makes the need for building very high-speed switches urgent. There are two major approaches in constructing a packet switch (Karol et al., 1987; Tamir and Frazier, 1988), input-queued and output-queued. We assume that packets are of the same size and each output can receive up to N packets in each time slot. Since packets are switched to outputs as soon as they arrive at inputs, input queues are not needed. Obviously output-queued switch provides the best delay-throughput performance. But its switch fabric and output buffer must operate N times faster than the line rate. This N-time speedup requirement makes output-queued switch difficult to scale.

On the other hand, an input-queued switch only allows one packet to be sent/received by each input/output in each time slot. No speedup is required. This makes input-queued architecture more suitable for high-speed implementation. Input-queued switches suffer from the well known problem of Head-of-Line (HOL) blocking. This limits the maximum throughput of an input-queued switch to just 58.6% under uniform traffic (Karol et al., 1987). To eliminate the HOL blocking, Virtual Output Queue (VOQ) is proposed (Tamir and Frazier, 1988), where each input port maintains a separate queue for each output (Fig. 1). A centralized scheduler is needed to maximize the throughput of a VOQ switch. The scheduling problem is equivalent to the matching problem in a bipartite graph (Chartrand, 1985). It is found that the Maximum Weight Matching (MWM; McKeown et al., 1996) algorithm can guarantee 100% throughput for any admissible traffic pattern. However, MWM algorithm has a high time complexity of O(N3).

Maximal Size Matching (MSM) algorithms with lower computation overheads are then proposed. A matching is of maximal size if no input or output is left unnecessarily idle (Anderson et al., 1993). A MSM algorithm is faster because finding a maximal size matching does not involve backtracking. Various efficient implementations of MSM are then designed. Among them, the approach of using iterative scheduling algorithms is widely adopted due to the use of massive parallel processing (Anderson et al., 1993; McKeown, 1999; McKeown, 1995; Li et al., 2000; Damm et al., 2001; Leonardi et al., 2001).

In general, an iterative scheduling algorithm consists of three phases, request, grant and accept. In the request phase, input ports send matching requests to output ports. In the grant phase, each output port grants at most one request. In the accept phase, each...
In basic HRF, each input always gives the highest scheduling priority to its preferred output at each time slot. To be more specific, if input $i$’s preferred output $j$ is backlogged, i.e., $\text{VOQ}(ij) > 0$, input $i$ (only) sends a single request to output $j$ (with a special rank number of 0). Subsequently, output $j$ grants input $i$’s request and input $i$ accepts output $j$’s grant.

If input $i$’s preferred output $j$ is not backlogged, i.e., $\text{VOQ}(ij) = 0$, input $i$ ranks its non-empty VOQs according to the descending order of the queue size, where rank $= 1$ is assigned to the longest VOQ. Then for each backlogged output, a request is sent together with its rank number. Since there are at most $N$ rank levels for $N$ VOQs of an input port, we need $\log_2 N$ bits to encode the rank number.

If an output receives multiple requests, the request with the highest rank is selected for grant (where rank $= 0$ is considered highest). In case of a tie, the winner is selected randomly. It should be noted that because the requests came from different inputs, they may have the same rank but different VOQ sizes, and a request with lower rank may have longer VOQ than another request with higher rank. We will show that this rank-based arbitration is more effective than VOQ size based arbitration because the former jointly considers both size and weight of a match.

Similarly, if an input receives multiple grants, the grant with the highest rank is selected for packet transmission. In this case, the selected VOQ is also the VOQ with the longest size. This is because all grants are for VOQs at the same input port. Unlike iSLIP, single-iteration algorithm HRF does not need to record the pointer of VOQ that transmits packet successfully. Therefore, no bit is exchanged during the grant phase of HRF.

### 2.2. HRF with request compression 1 (HRF/RC1)

Although the basic HRF is a single-iteration algorithm, it is still desirable to further cut down the communication overhead, in particular, the size of request message. To fully encode the rank information, $\log N$ bits are required for each request. We observe that in most cases, the fine granularity of rank information is not needed. We thus propose to encode only the most important rank information. Specifically, we only distinguish two rank levels, where a 1-bit request message suffices:

- Longest VOQ (with rank $= 1$, or binary “1”).
- All other non-empty VOQs (with rank $= 2$, or binary “0”).

Note that in the request message, we do not need to explicitly reserve rank $= 0$ to indicate a request to the preferred output. This is because at each time slot, each output port $j$ can identify its preferred input $i$ in advance based on the well-known relationship specified in (1). If output port $j$ receives a request from its preferred input $i$, no matter what rank number is in the message, this request is regarded as the highest priority rank $= 0$. In doing so, we honor the preferential input–output pair without imposing any communication overhead. The grant operation at an output is the same as the basic HRF, i.e., the request with the highest rank (including the inferred rank $= 0$) is selected for grant. Accordingly, the grant with the highest rank is selected by the input for packet transmission. For convenience, we denote this single-bit request variant of HRF as HRF/RC1.

Besides communication overhead, HRF/RC1 also reduces the computation complexity of HRF. In HRF, up to $N$ rank levels should be distinguished. A rather complicated sorting operation for all $N$ VOQs may be required. But HRF/RC1 only maintains 2 rank levels (excluding rank $= 0$), non-empty and longest. As such, each input port only needs to identify the longest VOQ.
We can generalize our request compression technique above by adopting a b-bit request message. In this case, we can distinguish $2^b$ rank levels. We call such a generalized scheme HRF/Rcb. The parameter $b$ can be tuned from 1 (HRF/RC1) to $\log N$ (basic HRF) to trade more communication overhead for better switch performance. But from the simulation results in Section 4, it is interesting to note that the performance gap between HRF/RC1 (1 bit) and basic HRF ($\log N$ bits) is very small. This verifies our statement that the fine granularity of rank information is not required and we only need to encode the most important rank information.

3. Related work on iterative algorithms

3.1. Multi-iterations algorithms

Based on the number of iterations to be executed, an iterative scheduling algorithm can be multi-iteration or single-iteration. Multi-iteration scheduling algorithms can be further divided into two types, multi-request (Anderson et al., 1993; McKeown, 1999, 1995) and single-request (Li et al., 2000; Damm et al., 2001; Leonardi et al., 2001). In a multi-request scheme, an input sends requests for all its non-empty VOQs. If an output/input port receives more than one request/grant, different contention resolution mechanisms are adopted by different scheduling algorithms. PIM (Anderson et al., 1993) picks up the winner randomly. iSLIP (McKeown, 1999) selects the winner according to a round robin pointer. iLQF (McKeown, 1995) prefers the longest VOQ and iOCF (McKeown, 1995) likes the VOQ with the oldest packet. Note that in the accept phase of the last iteration, an input port does not need to notify the accepted output as there is no subsequent iteration.

If an input port is allowed to send only a single request in the request phase, the communication overhead can be reduced. DRR (Li et al., 2000), pDRR (Damm et al., 2001) and LQD (Leonardi et al., 2001) are some interesting algorithms falling into this category. Specifically in each iteration, DRR selects one VOQ to send request according to a round robin pointer. The grant and accept phases are same as iSLIP. The only difference between DRR and pDRR is that multiple priorities are maintained in pDRR. LQD issues one request based on a probability proportional to the VOQ length. If an output port receives several requests, LQD grants one with equal probability. Due to the single request sent per input port, each input receives at most one grant. Since this single received grant will be accepted for sure, the request–grant–accept mode is degenerated into request-grant in all iterations of single-request algorithms.

Despite the fact that multi-iteration scheduling algorithm produces the maximal size match, it is not scalable (Mneimneh, 2008; Chen et al., 2009; Scicchitano et al., 2007).

3.2. Single-iteration algorithms

In single-iteration algorithm π-RGA (Mneimneh, 2008), there is a timer $C(i,j)$ to record the last time when VOQ($i,j$) transforms from empty to non-empty. Assumed that at time slot $t$, packet from VOQ($i,k$) is served/transmitted, then at time slot $t+1$, input port $i$ differentiates all non-empty VOQs to strong or weak based on the following rules. If $C(i,j) \leq C(i,k)$, then VOQ($i,j$)$\in$ {strong} and otherwise VOQ($i,j$)$\in$ {weak}. After such classification, if [strong] is empty but [weak] not, remove all VOQs in [weak] to [strong]. In the request phase, each input issues requests for all its non-empty VOQs. The VOQ status [strong] and [weak] and the value of $C(i,j)$ are indicated in every request. In the grant phase, priority is given to strong requests. If two or more strong requests arrive at an output port, the one with smallest $C(i,j)$ is granted. If there are only weak requests, the one with smallest $C(i,j)$ is chosen. In the accept phase, the VOQ with smallest $C(i,j)$ is accepted. π-RGA can achieve high throughput performance, especially under non-uniform traffic, but with much big request message size.

In SRA (Chen et al., 2009), each output port $j$ maintains a single FIFO status queue for all non-empty VOQ($i,j$)$\in$ {0,1,...,N−1} destined to $j$. Output $j$ always chooses the head of line from the status queue, say VOQ($i,j$), to send a grant. Output $j$ then removes VOQ($i,j$) to the tail of the status queue. Upon receiving multiple grants, an input port accepts all of them. SRA can find a maximum input/output matching in a single iteration, but it allows one input port to send multiple packets to different outputs during a single time slot. The associated speedup requirement is hard to implement.

3.3. Relationship with SRR

SRR (Scicchitano et al., 2007) is a single-iteration single-bit request scheme. For any input port $i$, the preferred request to output port $j$ is also calculated by a pre-determined and periodic sequence of $N$ patterns. If the preferred VOQ is empty, then the longest one is selected for sending a request. Each output $j$ also has a preferential input $i$ to grant based on the same sequence of $N$ patterns. If the preferred input request does not arrive, one request is randomly selected to grant.

Both SRR and our HRF/RC1 are single-iteration single-bit request algorithms. They claim the same communication overhead. Both of them give the scheduling priority to the preferred input–output pair first. If an input does not have backlogged packets for its preferred output, it sends requests to other outputs. In essence, "two" iterations are carried out in parallel, where the "first" iteration is for the preferential input–output pair and the "second" iteration is for the unmatched ports after preferential matching. This "2-iteration" nature makes HRF/RC1 and SRR superior to other single-iteration algorithms.

The major differences between HRF/RC1 and SRR are:

- SRR is designed for the scenario that the schedulers are fully distributed at each input/output port. But our HRF/RC1 applies to a single centralized scheduler. Therefore in case of large Round Trip Times (RTTs) for requests and grants, SRR yields a higher throughput. Otherwise, HRF/RC1 would provide the much better delay and throughput performance than SRR (please see the simulation results in Section 4).
- But please note that all non-empty VOQs in HRF/RC1 send the requests in parallel. In terms of time for sending requests, HRF/RC1 claims the same communication overhead as SRR.
- SRR is a two-phase algorithm composed of request and grant phases. HRF/RC1 requires the third phase accept to honor one grant per input port. But no bit is exchanged during the grant phase of HRF/RC1, which can be indeed regarded as a two-phase algorithm too.

3.4. Discussion

When multi-iteration algorithm iLQF (McKeown, 1995) executes one iteration only, it still requires more communication overhead (for carrying VOQs size) than single-iteration single-bit request algorithm HRF/RC1. But HRF/RC1 can provide a higher throughput than iLQF (please see the simulation results in Section 4). This is because HRF/RC1 jointly considers both size and weight of a match.
Note that no matter how many requests received, each output can only send a single grant per iteration. Then a single-iteration algorithm can just generate at most \( N \) grants. An efficient single-iteration algorithm should (a) give grant to heavily weighted request, and (b) avoid wasting grants in the interest of matching size.

If using VOQ size based arbitration (e.g. \( \pi \)-RGA and iLQF), output port \( j \) honors the request coming from input port \( i \), where VOQ\((i)\) has the maximum weight among all requests received by \( j \). But this grant would not be accepted yet if input \( i \) receives a more weighted grant from another output. In this case, the precious grant is wasted by trying to grant a more weighted request. In other words, VOQ size based arbitration seeks more matching weight but sacrifices the matching size of single-iteration algorithm. For instance in \( \pi \)-RGA (Mneimneh, 2008), assumed request from strong VOQ\((k)\) has the minimal value of \( C(lk) \) among all requests to output port \( k \). So output port \( k \) gives the grant to input port \( l \). At the same time slot, input port \( l \) also receives another grant from output port \( h \). Note that even \( C(lk) \) is the minimal one in output \( k \) and it is possible that \( C(lk) > C(lh) \), since \( C(lh) \) is granted by another output \( h \). Then input port \( l \) accepts output port \( h \) and ignores the grant from output port \( k \). Consequently, the grant from output port \( k \) is wasted and the size of matching is decreased.

Unlike VOQ size based arbitration, the rank-based arbitration (i.e. HRF/RC1) grants the request from input port \( i \) where VOQ\((i)\) has the maximum weight among input \( i \). Then output port \( j \) actually issues a grant to the input that will accept output \( j \)'s grant most likely. For example in HRF/RC1, if an output grants a request with rank\(=1 \), such grant is predestined to be accepted for sure, since rank\(=1 \) presents the longest VOQ at an input port. Therefore, HRF/RC1 gives grant to the most weighted request (among an input) and also avoids wasting grants as best as it can. As compared with VOQ size based arbitration, the rank-based arbitration yields the similar matching weight but more matching size in a single-iteration algorithm. It is interesting to note that under multi-iteration scenario, VOQ size based arbitration can provide the similar performance as the rank-based. This is because the matching size sacrificed by size based in the first iteration can be mended during the subsequent iterations. Nevertheless, as far as the single-iteration case, VOQ rank-based arbitration outperforms size based.

In summary, the “2-iteration” nature and rank-based arbitration make HRF/RC1 superior to other single-iteration algorithms. In Appendix A, we provide some analytical results for HRF/RC1. Specifically, it guarantees 100% throughput if the traffic arrival rate of each flow is no larger than \( 1/N \) packets/slot. Moreover, we also prove that HRF/RC1 satisfies the max–min fairness criterion.

4. Performance evaluations

In this section, we study the performance of HRF/RC1 by simulations. Since HRF/RC1 is a single-iteration algorithm, it is only fair to compare it with other single-iteration algorithms and some multi-iteration algorithms executing one iteration:

(a) SRR (Sciccitano et al., 2007), single-iteration algorithm.
(b) \( \pi \)-RGA (Mneimneh, 2008), single-iteration algorithm.
(c) iSLIP (McKeown, 1999), multi-iteration algorithm executing one iteration.
(d) iLQF (McKeown, 1995), multi-iteration algorithm executing one iteration.
(e) Output-queued switch, served as a lower bound.

Further as a single-bit request algorithm, our HRF/RC1 only competes with SRR and (1-iteration) iSLIP. The basic HRF and HRF/RC2 are also implemented for reference. We present simulation results for switch with size \( N=32 \) below, but the same conclusions and observations apply for other sizes.

4.1. Uniform traffic

Uniform traffic is generated as follows. At every time slot for each input, a packet arrives with probability \( p \) (input load \( p \)) and destines to each output with equal probability. From Fig. 2, we can see that our HRF/RC1 can obtain up to 100% throughput. This is consistent with the analytical result in Appendix A, i.e. when \( \rho = 1/N \) (for all \( i, j=0, 1,\ldots,N-1 \)), HRF and its variants can
achieve 100% throughput. Notably, the performance curves of HRF/RC1, HRF/RC2 and the basic HRF almost overlap with each other. This verifies that the fine granularity of the rank information is not needed. Compared with SRR and iSLIP, HRF/RC1 gives significantly smaller delay if \( p \) is reasonably large (> 0.6). When \( p = 0.8 \), SRR requires 41.5 time slots, and HRF/RC1 only 10.9, cutting down the delay by more than 3 times.

4.2. Uniform bursty traffic

Bursty arrivals are modeled by the ON/OFF traffic model, which is a special instance of the two-state Markov-modulated process (Hu and Yeung, 2010). In the ON state, a packet arrival is generated in every time slot. In the OFF state, there are no packet arrivals. Packets of the same burst have the same output and the output for each burst is uniformly distributed. Given the average input load of \( p \) and average burst size \( s \), the state transition probabilities from OFF to ON is \( p/[s(1-p)] \) and from ON to OFF is \( 1/s \). Without loss of generality, we set burst size \( s = 30 \) packets.

From Fig. 3, we can see that delay builds up quickly with input load. iSLIP only achieves 63% throughput but HRF/RC1 obtains 87% throughput. When \( p > 0.7 \), there is a tiny performance gap between HRF/RC1 and HRF. Nevertheless, HRF/RC1 consistently shows superior to SRR even when traffic is light. For example at

![Fig. 3. Delay vs input load, under uniform bursty traffic.](image)

![Fig. 4. Delay vs input load, under hot-spot traffic.](image)
p = 0.6, with SRR packets experience an average delay of 200 time slots, whereas for HRF/RC1 it is just 61.4. This can be attributed to the more matching size and weight obtained by HRF/RC1. We can also see that HRF/RC1 outperforms π-RGA if p ≤ 0.7. When p > 0.7, π-RGA can catch the bursty flows so it shows the less delay than HRF/RC1. But it should be noted that HRF/RC1 is a single-bit request algorithm while π-RGA is not scalable/practical because of its heavier communication overhead.

4.3. Hot-spot traffic

We assume packets arriving at each input port in each time slot follow the same independent Bernoulli process with probability p. Hot-spots are generated as follows (Hu and Yeung, 2010; Hu et al., 2012). For input port i, packet goes to output i + N/2 mod N with probability 0.5, and goes to other outputs with the same probability 1/[2(N – 2)]. From Fig. 4, again we can see that SRR only achieves 76% throughput but HRF/RC1 provides 86% throughput. HRF/RC1 consistently shows superior to SRR and iSLIP. Again, this is because HRF/RC1 jointly considers both size and weight of a match.

In summary for all three classic traffic patterns, HRF/RC1 yields the best delay-throughput performance among the single-iteration scheduling algorithms with the comparable communication overhead.

5. Conclusions

In this paper, we proposed a single-iteration scheduling algorithm for input-queued switches, named Highest Rank First (HRF). HRF always gives the highest priority to N distinct input-output pairs, which vary in each time slot according to a pre-determined and periodic sequence of N patterns. When the preferred VOQ(i,j) is empty, input i sends a request with a rank number to each backlog output. This rank number is maintained at each VOQ according to the descending order of the queue size. The higher priority is given to the higher rank in issuing grants and acceptances. To further cut down communication overhead, we compressed the request message size in HRF by reducing the granularity of the rank levels. This produced an effective single-iteration single-bit request scheduling algorithm, named HRF/RC1. The simulation results showed that HRF/RC1 provides the best delay-throughput performance among all existing single-iteration single-bit request scheduling algorithms.

Acknowledgments

This work was supported in part by the Small Project Funding (The University of Hong Kong, No. 201007176210), The Zhejiang Province Welfare Applied Research Program (No. 2010C31071), Fundamental Research Funds for the Central Universities (No. 2010QNA5032 & 2012QNA5016), National Science and Technology Major Project (No. 2011ZX03003-003-03).

Appendix A

A.1. Constructing fluid model

Let $\lambda_g$ be the mean packet arrival rate to VOQ(i,j). A traffic pattern/matrix is admissible if

$$\sum_i \lambda_{ij} \leq 1, \sum_j \lambda_{ij} \leq 1.$$ (2)

In other words, there are no oversubscribed inputs and outputs.

We follow the approach in Berger (2006) and Dai and Prabhakar (2000) and first establish a fluid model for the basic Highest Rank First (HRF) and its variants. Let the number of packets in VOQ(i,j) at the beginning of time slot n be $Z_g(n)$. Let the cumulative number of arrivals and departures for VOQ(i,j) at the beginning of slot n be $A_g(n)$ and $D_g(n)$, respectively. We have:

$$Z_g(n) = Z_g(0) + A_g(n) - D_g(n), n \geq 0, i,j = 0, \ldots, N-1.$$ (3)

Assume that the packet arrival process obeys the strong law of large numbers with probability one, i.e.

$$\lim_{n \to \infty} \frac{A_g(n)}{n} = \mu_{ij}, \quad i,j = 0, \ldots, N-1.$$

The switch is, by definition, rate stable if:

$$\lim_{n \to \infty} \frac{D_g(n)}{n} = \mu_{ij}, \quad i,j = 0, \ldots, N-1.$$ (4)

If a switch is rate stable for an admissible traffic matrix, then the switch delivers 100% throughput.

The fluid model is determined by a limiting procedure illustrated below. Firstly, the discrete functions are extended to right continuous functions. For arbitrary time $t \in [n, n+1)$:

$$A_g(t) = A_g(n),$$

$$Z_g(t) = Z_g(n),$$

$$D_g(t) = D_g(n) + (t-n)(D_g(n+1) - D_g(n)).$$

Note that all functions are random elements of $\mathbb{D}[0, \infty)$. We shall sometimes use the notation $A_g(\cdot, \omega), Z_g(\cdot, \omega)$ and $D_g(\cdot, \omega)$ to explicitly denote the dependency on the sample path $\omega$. For a fixed $\omega$, at time $t$, we have (Berger, 2006):

$$A_g(t, \omega),$$

$$Z_g(t, \omega),$$

$$D_g(t, \omega),$$

the cumulative number of arrivals to VOQ(i,j),

the number of packets in VOQ(i,j),

and the cumulative number of departures from VOQ(i,j).

For each $r > 0$, we define

$$\bar{A}_g(t, \omega) = r^{-1}A_g(rt, \omega),$$

$$\bar{Z}_g(t, \omega) = r^{-1}Z_g(rt, \omega),$$

$$\bar{D}_g(t, \omega) = r^{-1}D_g(rt, \omega).$$

It is shown in Dai and Prabhakar (2000) that for each fixed $\omega$ satisfying (3) and any sequence $\{r_n\}$ with $r_n \to \infty$ as $n \to \infty$, there exist a subsequence $\{r_n\}$ and the continuous functions $\bar{A}_g(\cdot, \omega), \bar{Z}_g(\cdot, \omega), \ldots$, where $\bar{A}_g(t, \omega), \bar{Z}_g(t, \omega), \ldots$ converges to uniformly on compacts as $k \to \infty$ for any $t \geq 0$:

$$\bar{A}_g(\cdot, \omega) \to \lambda_{ij} t,$$

$$\bar{Z}_g(\cdot, \omega) \to Z_g(t),$$

$$\bar{D}_g(\cdot, \omega) \to D_g(t).$$ (4)

Definition 1. Any function obtained through the limiting procedure in (4) is said to be a fluid limit of the switch. So the fluid model equation using HRF and its variants is

$$\bar{Z}_g(t) = \bar{Z}_g(0) + \lambda_{ij} - \bar{D}_g(t), \quad t \geq 0.$$ (5)

Definition 2. The fluid model of a switch operating under a scheduling algorithm is said to be weakly stable if for every fluid model solution $(\bar{D}, \bar{Z})$ with $\bar{Z}(0) = 0, \bar{Z}(t) = 0$ for almost every $t \geq 0$. From Berger (2006), the switch is rate stable if the corresponding fluid model is weakly stable. Our goal here is to prove that for every fluid model solution $(\bar{D}, \bar{Z})$ using HRF, $\bar{Z}(t) = 0$ for almost every $t$. To prove $\bar{Z}(t) = 0$, we will use the following Fact 1 from Dai and Prabhakar (2000):

Fact 1. Let $f$ be a non-negative, absolutely continuous function defined on $\mathbb{R}^+ \cup \{0\}$ with $f(0) = 0$. Assume that for almost every time $t$, $f(t) > 0$ and $f(t) \leq 0$. Then $f(0) = 0$ for almost every $t \geq 0$.  

Note that $\mathbb{R}^+$ is the set of positive real numbers, and $f'(t)$ denotes the derivative of function $f$ at time $t$.

### A.2. 100% throughput proof

In the following, we show that HRF and its variants are stable if all $\lambda_q \leq 1/N$.

**Theorem 1.** (Sufficiency) When $\lambda_q \leq 1/N$ (for all $i, j=0, 1, \ldots N-1$), HRF and its variants can achieve 100% throughput for any admissible traffic pattern obeyed the strong law of large numbers.

**Proof.** Define $B \triangleq \{ m : Z_{im}(t) > 0 \}$. Let $G_i(t)$ denote the joint queue occupancy of all non-empty VOQs at input port $i$:

$$G_i(t) = \sum_{m \in B} Z_{im}(t)$$

(6)

$Z(t)$ is a non-negative, absolutely continuous function, so $G_i(t)$ is also non-negative and absolutely continuous. We can see that $G_i(0)=0$ and then the derivative of $G_i(t)$ is

$$G_i(t) = \sum_{m \in B} Z_{im}(t)$$

Combine the above equation with (5), we get

$$G_i(t) = \sum_{m \in B} \lambda_{im} - \sum_{m \in B} D_{im}(t)$$

(7)

From the admissible traffic condition (2) and $\lambda_q \leq 1/N$ (for all $i, j=0, 1, \ldots N-1$),

$$G_i(t) \leq \frac{h}{N} - \sum_{m \in B} D_{im}(t)$$

(8)

where $h = \| B \| \geq 0$.

Suppose that $G_i(t) > 0$. This implies that for $m_1 \in B$, $\forall m_2 \notin B$, $Z_{im}(t) > 0$ and $Z_{im}(t) = 0$. Then $Z_{im}(t) - Z_{im}(t) > 0$. By the continuity of these functions, $\exists$ such that

$$\min_{t \in [t_{i+1}]} (Z_{im}(t') - Z_{im}(t')) > 0 \quad \forall m_1 \in B, \forall m_2 \notin B$$

Let

$$q = \min_{m_1 \in B} \min_{t \in [t_{i+1}]} (Z_{im}(t') - Z_{im}(t'))$$

For a large enough $k$, we have $Z_{im}(t') = Z_{im}(t') \geq q/2$, where $t' \in [t_i, t_{i+1}]$ $\forall m_1 \in B$ and $\forall m_2 \notin B$. Also for large enough $k$, we have $Z_{im}(t') = Z_{im}(t') \geq 1$, where $t' \in [r_n, r_n + (t + \delta)]$, $\forall m_1 \in B, \forall m_2 \notin B$. This means that in the long time interval $[r_n, r_n + (t + \delta)]$, any non-empty VOQ at input port $i$ belongs to the set $U \triangleq \{ VOQ(l,m) : m \in B \}$ and any VOQ that belongs to set $U$ is non-empty (Berger, 2006). Since HRF and its variants always give the highest priority to the preferred input-output pair calculated by (1), each non-empty VOQ sends at least one packet per N time slots during the same interval. Then in the long time interval $[r_n, r_n + (t + \delta)]$, input port $i$ sends at least $h = \| B \| \cdot N$ packets per N time slots,

$$\sum_{m \in B} D_{im}(r_{n} t') - D_{im}(r_{n} t) \geq Lh$$

(8)

where $L \in Z, N L \leq r_n t' - r_n t \leq N L + N$. So we have

$$L \geq \frac{r_n (t' - t)}{N} - 1$$

(9)

Combine (8) with (9),

$$\sum_{m \in B} D_{im}(r_{n} t') - D_{im}(r_{n} t) \geq \frac{h r_n (t' - t)}{N} - h$$

(10)

Further dividing the above equation by $(t' - t)$, and letting $t' \to t$, the derivative of the fluid limit is

$$\sum_{m \in B} D_{im}(t) \geq \frac{h}{N}$$

(11)

Then combine (7) and (10), we get

$$G_i(t) \leq 0$$

Based on Fact 1, $G_i(t) \to 0$ for almost every $t \geq 0$. Due to (6) and $G_i(t) \to 0$, then $Z_{im}(t) = 0$ for almost every $t \geq 0$. We proved Theorem 1.

### A.3. Max–min fairness criterion

In the following, we prove that HRF and its variants also satisfy the max–min fairness criterion. Recall that in HRF each input prefers each output exactly once in every $N$ slots. Accordingly, each non-empty VOQ can send at least one packet per $N$ slots. But in iSLIP, a non-empty VOQ is served for sure only after $N^2$ slots (McKeown, 1999). Intuitively, HRF and its variants provide fairer resource allocation. We first borrow two definitions (Hosagrahara and Sethu, 2008):

**Definition 3.** The allocation vector $\{ u_i \}$ is said to be feasible if and only if:

1. Each entity receives an allocation greater than or equal to zero; that is, for all $i, u_i \geq 0$.
2. The total allocated resource is less or equal to the available resource $U$; that is, $\sum u_i \leq U$.

**Definition 4.** For the demand vector $\{ v_i \}$, the allocation vector $\{ u_i \}$ is said to be max–min fair if:

1. It is feasible.
2. No entity receives an allocation greater than its demand; that is, for all $i, u_i \leq v_i$.
3. For all $i$, the allocation of entity $i$ cannot be increased while satisfying the above two conditions and without reducing the allocation of some other entity $j$ for which $u_j \leq u_i$.

As long as an algorithm meets the three conditions above, it satisfies the max–min fairness criterion (Hosagrahara and Sethu, 2008). Note that in HRF, the demand vector $\{ v_i \}$ is the traffic load from input $i$ to output $j$. Let the capacity of output $j$ be $U$, i.e. the available resource is $U$. Assume HRF and its variants allocate $U$ to each input $i$ with allocation $u_i$ ($i=0, 1, \ldots N-1$). Obviously, $u_i \geq 0$ and $\sum u_i \leq U$ ($i=0, 1, \ldots N-1$). So $u_i$ is feasible (condition 1). As long as a VOQ($ij$) is empty (no matter it is preferred or not), no matching request is sent from VOQ($ij$). Therefore in HRF for all $i$, $u_i \leq v_i$ can be ensured (condition 2).

In the following, we focus on condition 3, where we increase some bandwidth allocation $u_i$ and see how this would affect other inputs. Assume the switch has been “warmed up”. Let $c_i$ be the number of requests sent by VOQ($ij$) during $L$ time slots. We have

$$c_i \leq L$$

(11)

If $c_i$ of VOQ($ij$) is larger than $c_k$ of VOQ($kj$), according to HRF, input $i$ will get a larger share of output $j$'s bandwidth in a long
term. That is
\[ u_i \geq u_k \quad \text{if } c_i \geq c_k \tag{12} \]

By conditioning on the value of \( c_i \), two cases are considered:

- \( c_i < L \): In one or more time slots, VOQ(\( i,j \)) does not send request because it is empty. Then traffic load \( v_i \) is satisfied by allocation \( u_i \), i.e.
\[ \lim_{L \to \infty} c_i / L = v_i = u_i \]

Therefore, \( u_i \) cannot be further increased because \( u_i \) is bounded by \( v_i \) in condition 2.

- \( c_i = L \): VOQ(\( i,j \)) always sends request during the whole \( L \) time slots, since it is invariably occupied. This indicates that traffic load \( v_i \) cannot be satisfied by allocation \( u_i \) because output \( j \) is fully loaded:
\[ \sum u_i = U \tag{13} \]

From (11), we have:
\[ c_i \geq c_k \quad \text{for all } k = 0, 1, \ldots, N-1 \]

Combine it with (12), we get:
\[ u_i \geq u_k \quad \text{for all } k = 0, 1, \ldots, N-1 \tag{14} \]

To increase \( u_i \), we have to reduce some \( u_k \) due to (13). Then we reduce the allocation to some input \( k \) for \( u_i \geq u_k \) (14), which proves that condition 3 is ensured.

Combining the proof for all the three conditions in Definition 4, we proved that HRF and its variants satisfy the max–min fair criterion.

References
