Fair Scheduling in Input-Queued Switches under Inadmissible Traffic

Neha Kumar, Rong Pan, Devavrat Shah
Departments of EE & CS
Stanford University
{nehak, rong, devavrat}@stanford.edu

Abstract—**In recent years, several high-throughput low-delay scheduling algorithms have been designed for input-queued (IQ) switches, assuming admissible traffic. In this paper, we focus on queueing systems that violate admissibility criteria.**

We show that in a single-server system with multiple queues, the Longest Queue First (LQF) policy disallows a fair allocation of service rates [1]. We also describe the duality shared by LQF's rate allocation and a fair rate allocation. In general, we demonstrate that the rate allocation performed by the Maximum Weight Matching (MWM) scheduling algorithm in overloaded IQ switches is unfair. We attribute this to the lack of coordination between admission control and scheduling, and propose fair scheduling algorithms that minimize delay for non-overloaded queues.

Keywords— **Congestion Control, Quality of Service and Scheduling, Stochastic Processes and Queueing Theory, Switches and Switching, Resource Allocation**

I. Introduction

The input-queued (IQ) switch architecture is widely used in high-speed switching. This is due to its low memory bandwidth requirements compared to those of output-queued and shared-memory architectures, making it the preferred choice.

In an $N \times N$ IQ switch, we assume fixed-size cells (packets). Each input has N FIFO virtual output queues (VOQs), one for each output [2]. Packets queue up at the inputs, arriving at input i for output j at an average rate λ_{ij}. In each time slot, at most one packet can arrive at each input and at most one can be transferred to an output. Consider these conditions for $\Lambda = [\lambda_{ij}]$:

$$\sum_{j=1}^{N} \lambda_{ij} < 1, \ \forall i \qquad \sum_{i=1}^{N} \lambda_{ij} < 1, \ \forall j$$

The first condition is enforced by the line-rate constraints at the inputs. Incoming traffic is called *admissible* when the second condition is satisfied and *inadmissible* otherwise. The switch scheduling problem reduces to a matching problem in a weighted bipartite graph with N inputs and N outputs [3].

A. Background and Motivation

The primary performance metric of an IQ switch scheduling algorithm is the throughput it delivers. The Maximum Weight Matching (MWM) algorithm has been shown to achieve 100% throughput when arriving traffic is admissible and obeys the Strong Law of Large Numbers. Practical heuristics to approximate MWM [7] [4] too have been proposed. In a single-server queueing system, Longest Queue First (LQF) is also known to achieve 100% throughput. These results focus on admissible traffic conditions however, when in practice traffic is frequently inadmissible. Herein lies our motivation for this paper, where we study scheduling policies for *inadmissible* traffic conditions in IQ switches.

Under admissible traffic, stable scheduling algorithms grant every flow its desired service, and there does not arise a need for fairness in rate allocation [4]. Under inadmissible traffic, not all flows can receive desired service. We observe the rate allocations performed by LQF and MWM in such a scenario, and prove that they lack fairness. This motivates our search for a scheduling policy that performs a fair rate allocation, given inadmissibility. We now summarize our results.

B. Outline and Results

In section II, we formalize the notion of fairness that we will use. We then commence our study of IQ switch-scheduling under inadmissible traffic conditions by observing the performance of LQF in a system with multiple queues and an over-subscribed server in section III. As an interesting side observation, we also present the duality shared by the rate allocations determined by LQF and Max-Min fairness. In section IV, we provide Fair-LQF, an algorithm that incorporates fairness into LQF to perform a provably fair rate allocation.

In section V, we extend our study to the $N \times N$ switch and observe the rate allocation performed by MWM. We show that it lacks fairness, and propose the Fair-MWM algorithm in section VI, conjecturing that this too performs a fair rate allocation. Our conclusions follow in section VII.

II. Fairness in Scheduling

To define a fair rate allocation for flows, we use the established notion of *Max-Min fairness* [1].

Definition 1 (Max-Min Fairness) Let there be n flows arriving at a server of capacity C with rates $\lambda_1, \cdots, \lambda_n$ respectively. A rate allocation $r = (r_1, \cdots, r_n)$ is called Max-Min fair iff
(i) $\sum_n r_i \leq C, \ r_i \leq \lambda_i$, and
(ii) any r_i can be increased only by reducing r_j s.t. $r_j \leq r_i$.

Definition 2 (Fairness in a Switch) There exist N^2 flows in an $N \times N$ switch. We call $R = [r_{ij}]$ a fair allocation for $\Lambda = [\lambda_{ij}]$ iff it is Max-Min fair for every output.

[1] The formal definition of fairness is provided later.

[2] The Virtual Output Queueing architecture improves performance by preventing Head-of-Line blocking [5].

[3] In this paper, we take the weight of the edge (i, j) as the size of VOQ_{ij}.

[4] In this paper, each VOQ defines one flow. This can easily be generalized to accommodate the arrival of multiple flows at an input that are intended for the same output.

III. Longest Queue First and Fairness

We first consider a system of N queues and a server of unit capacity. The arrival rate for queue i, $1 \leq i \leq N$, is denoted by λ_i, and the cumulative number of arrivals until time n is denoted by $A_i(n)$, where $A_i(0) = 0$. We assume that the arrivals obey the Strong Law of Large Numbers (SLLN) that states:

$$\lim_{n\to\infty} \frac{A_i(n)}{n} = \lambda_i \ \forall i \ \ w.p.1 \quad (1)$$

Let $Q_i(n)$ denote the size of queue i at time n, $D_i(n)$ the number of departures from queue i until time n ($D_i(0) = 0$), and $T_i(n)$ the number of time slots queue i is scheduled for service in $[0, n]$. LQF always serves the longest queue, breaking ties arbitrarily. For $n \geq 0$, the equations below describe queue sizes, departures and the busyness of the server over time.

$$\begin{aligned} Q_i(n) &= Q_i(0) + A_i(n) - D_i(n) \\ D_i(n) &= D_i(n-1) + (T_i(n) - T_i(n-1))(Q_i(n-1) > 0) \end{aligned}$$

$$T_i(\cdot) \text{ is non-decreasing and } \sum_{i=1}^{N} T_i(n) = n$$

The dynamics of the system are completely described by $S(n) = (Q_i(n), D_i(n), T_i(n))_{i=1}^N$.

A. Fluid Model for a Queue

We employ the *fluid model* technique to study LQF [3]. To do this, we define a sequence of systems indexed by $r = 1, 2, 3 \ldots$ with the associated description vectors $S^r(t) = (Q_i^r(n), D_i^r(n), T_i^r(n))_{i=1}^N$. The relationship between $S^r(t)$ and $S(\cdot)$ is described by:

$$S^r(t) = S(\lfloor rt \rfloor)/r$$

The fluid model solution of the system is characterized by $\bar{S}(t)$ such that:

$$\bar{S}(t) = \lim_{r\to\infty} S^r(t)$$

$S^r(t)$ is a non-deterministic quantity. We would like to show that the limiting quantity $\bar{S}(t)$ is a deterministic solution to a set of differential equations. We proceed by showing the existence of a limiting quantity and its characterization.

If for all i, $|D_i^r(t) - D_i^r(t')| \leq |t - t'|$, $|T_i^r(t) - T_i^r(t')| \leq |t - t'|$, and $\lim_{r\to\infty} |A_i^r(t) - \lambda_i t| = 0$ then $\bar{S}(t)$ exists [2] and satisfies the fluid model equations [5] as follows:

$$\bar{Q}_i(t) = \bar{Q}_i(0) + \lambda_i t - \bar{D}_i(t) \quad (2)$$

$$\frac{d\bar{D}_i(t)}{dt} = \frac{d\bar{T}_i(t)}{dt} \text{ if } \bar{Q}_i(t) > 0 \quad (3)$$

$$\bar{T}_i(\cdot) \text{ is non-decreasing and } \sum_{i=1}^{N} \bar{T}_i(t) = t \quad (4)$$

We can check that our system satisfies these conditions if we assume that arriving traffic satisfies condition (1).

By serving only the longest queue at all times, LQF further imposes the following condition on the evolution of $\bar{T}_i(\cdot)$:

$$\text{If } \exists j \text{ s.t. } \bar{Q}_i(t) < \bar{Q}_j(t) \text{ then } \frac{d\bar{T}_i(t)}{dt} = 0. \quad (5)$$

[5] For an example, see [3].

B. Rate Allocation under LQF

When traffic is inadmissible, queues that do not receive desired service grow unboundedly. Since LQF is a non-idling policy, it services a queue in every time slot. For large enough t, $r_i(t) = T_i(t)/t$ denotes the service rate allocated to queue i by LQF. The fluid model equations (2) – (5) characterize LQF's rate allocation for arrival rates $\lambda_1, \cdots, \lambda_n$.

For admissible traffic, techniques such as Lyapunov analysis may be used to show that LQF services every queue at its arrival rate. For inadmissible traffic, we state the following theorem to describe LQF's rate allocation:

Theorem 1: Let λ_i represent the arrival rate for queue i, $1 \leq i \leq N$. Assume $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_N \geq \lambda_{N+1} = 0$. Let $r_i(t)$ represent the rate allocated by LQF to queue i in time slot t. Then $\forall t \geq 0$:
(i) $0 \leq r_k(t) \leq \lambda_k$ s.t. $\sum_k r_k(t) = 1$
(ii) Let $1 \leq l \leq N$ be the smallest index such that:

$$\Delta \triangleq (\sum_{i=1}^{l} \lambda_i - 1)/l > \lambda_{l+1}$$

Then $\forall k, r_k(t) = (\lambda_k - \Delta)^+$. That is, all queues served at a positive rate grow at the rate Δ, while the rest grow at their respective arrival rates [6].

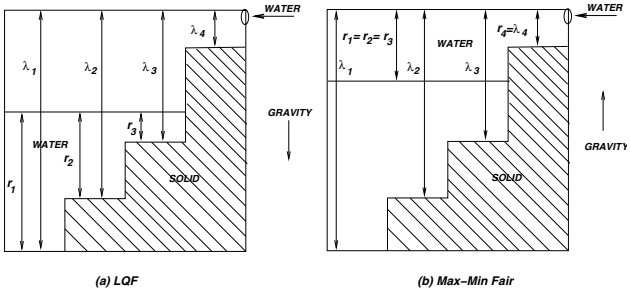

Fig. 1. Comparison of LQF and Max-Min Fairness.

C. LQF vs. Max-Min Fairness

We momentarily digress to present an interesting duality that exists between the rates allocated by LQF and those allocated by Max-Min fairness. It is illustrated in Figure 1, where the shaded area represents a solid surface and the rest is empty space. There are N steps such that step i has depth λ_i and unit width, i.e. there are N columns with volumes $\lambda_1, \cdots, \lambda_N$ respectively and column i represents queue i with arrival rate λ_i.

We begin our experiment by pouring water (of unit volume) through the hole. In Figure 1(a) gravity acts downwards, causing the water to fill the columns deepest-first. In Figure 1(b) gravity acts upwards, so that the shallowest column fills up first. When the pouring is complete, the water level in column i indicates the service rate allocated to queue i by LQF in 1(a) and Max-Min fairness in 1(b).

We observe that downwards-acting gravity causes the water to fill up the deepest column first, just as LQF services the longest queue at the expense of the less-demanding flows. Conversely, in Figure 1(b), the water first fills all columns equally. When it reaches a depth of λ_N, the additional water fills the remaining

[6] The proof can be found at http://simula.stanford.edu/∼neha/FMWM.pdf.

$N-1$ columns equally and so on. After the water has been poured, all partially filled columns have equal depth. Moreover, the water level in column i indicates the Max-Min fair rate allocation to queue i.

This experiment reveals the duality shared by the rates allocated by LQF and those allocated by Max-Min fairness. We now propose an algorithm that incorporates Max-Min fairness into LQF to provide a fair rate allocation.

IV. FAIR-LQF

The Fair-LQF algorithm combines the advantages of Max-Min fairness and LQF. Based on individual queue sizes and a pre-specified threshold value, it maintains a *congested* list of queues whose sizes exceed the threshold [7] and an *uncongested* list of all other queues. In the limiting case of this threshold going to ∞, a congested queue is one that is served at less than its desired rate.

A. The Algorithm

If n_c represents the number of congested queues and n_{uc} represents the number of non-empty, uncongested queues, for the next n_c time slots, Fair-LQF serves every congested queue exactly once. For the remainder n_{uc} time slots, it mimics LQF. The pseudocode for Fair-LQF is as follows:

```
//Information Collection
if (queue_size(queue_ID) >= threshold)
    add_to_congested_list(queue_ID);
// Scheduling
// Step1: scheduling congested ports
m = number-of-congested-queues;
while ( m != 0) {
  // Round-Robin
  schedule-congested-queues();
  m--;
}
//Step2: scheduling uncongested ports
m = number-of-nonempty-uncongested-queues;
while ( m != 0) {
  LQF-schedule-uncongested-queues();
  m-- ;
}
```

Fair-LQF deviates from LQF as it proactively limits the throughput of a congested queue. Under admissible traffic, Fair-LQF behaves identically to LQF, since the limiting size of the queues is 0. Under inadmissible traffic, it does a provably Max-Min fair rate allocation.

Theorem 2: Let $\lambda_1 \geq \cdots \geq \lambda_N$ be the arrival rates of N queues and the server capacity be 1. Let $R_1, \cdots, R_N$ denote the Max-Min fair rate allocation, and $r_1, \cdots, r_N$ denote the Fair-LQF rate allocation. Then $\forall k, r_k = R_k$, i.e. Fair-LQF is Max-Min fair [8][9].

B. Performance Evaluation

We studied the performance of LQF and Fair-LQF via simulations and compared it to Max-Min fairness. The simulations attest that the Fair-LQF rate allocation is Max-Min fair and provides lower delay for uncongested queues [10].

[7] The threshold may be chosen freely and does not affect the long-term performance of the algorithm.

[8] The proof can be found at http://simula.stanford.edu/∼neha/FMWM.pdf.

[9] This theorem holds regardless of the threshold value.

[10] The simulation results are at http://simula.stanford.edu/∼neha/FMWM.pdf.

V. MAXIMUM WEIGHT MATCHING AND FAIRNESS

We now extend our study to the $N \times N$ switch and observe MWM's rate allocation in an overloaded switch to show that it lacks fairness.

An input-output matching is represented by a permutation matrix $\pi = [\pi_{ij}]_{i,j \leq N}$ where $\pi_{ij} = 1$ iff it matches input i to output j. A scheduling algorithm $\mathcal{S}$ must determine a matching $\pi(n)$ for each time slot n.

The arrival rate for VOQ_{ij}, $1 \leq i, j \leq N$, is denoted by λ_{ij}, and the cumulative number of arrivals until time n is denoted by $A_{ij}(n)$, where $A_{ij}(0) = 0$. We assume that the arrivals obey the Strong Law of Large Numbers (SLLN) that states:

$$\lim_{n \to \infty} \frac{A_{ij}(n)}{n} \quad = \quad \lambda_{ij} \; \forall i, j \;\; w.p.1 \tag{6}$$

Let $Q_{ij}(n)$ denote the size of VOQ_{ij} at time n, $D_{ij}(n)$ the number of departures from VOQ_{ij} until time n ($D_{ij}(0) = 0$), and $W(n) = [W_{ij}(n)]$, where $W_{ij}(n)$ denotes the weight of the edge (i, j) at time n in the switch bipartite graph [11]. Then the weight of a matching π is defined thus:

$$W^{\pi}(n) = \sum_{i,j} \pi_{ij} W_{ij}(n) = \pi W(n)$$

Definition 3 (Maximum Weight Matching) A maximum weight matching algorithm is one that determines a matching $\pi^w(n)$ at time n such that:

$$\pi^w(n) \quad = \quad \arg\max_{\pi} \{W^{\pi}(n)\} \tag{7}$$

We now consider the dynamics of the discrete-time switch. Let MWM be our scheduling algorithm and $\prod$ be a collection of all possible matchings [12]. For any $\pi \in \prod$, let $T_{\pi}(n)$ be the cumulative amount of time that a permutation π is scheduled in the time interval $[0, n]$. Again, $T_{\pi}(0) = 0, \forall \pi \in \prod$. For $n \geq 0$, the equations below describe queue sizes, departures, and the busyness of the server over time:

$$Q_{ij}(n) \quad = \quad Q_{ij}(0) + A_{ij}(n) - D_{ij}(n) \tag{8}$$

$$D_{ij}(n) \quad = \quad \sum_{\pi \in \prod} \sum_{l=1}^{n} \pi_{ij} 1_{Q_{ij}(l) > 0} (T_{\pi}(l) - T_{\pi}(l-1)) \tag{9}$$

$$T_{\pi}(\cdot) \text{ is non-decreasing and } \sum_{\pi \in \prod} T_{\pi}(n) = n \tag{10}$$

A. Fluid Model for a Switch

We use the fluid model technique again to study MWM. By appropriately scaling the discrete-time switch, we can obtain the fluid model for a switch [3]. Based on equations (8) – (10), it has been shown [3] that under condition (6), the continuous, deterministic fluid model of a switch is as follows:

$$Q_{ij}(t) \quad = \quad Q_{ij}(0) + \lambda_{ij} t - D_{ij}(t) \geq 0, \; t \geq 0 \tag{11}$$

$$\dot{D}_{ij}(t) \quad = \quad \sum_{\pi \in \prod} \pi_{ij} \dot{T}_{\pi}(t), \text{ if } Q_{ij}(t) > 0 \; , \; t \geq 0 \tag{12}$$

[11] In this paper, we let $W_{ij}(n)$ be $Q_{ij}(n)$.

[12] For an $N \times N$ switch, $|\prod| = N!$.

$$T_\pi(\cdot) \text{ is non-decreasing, and } \sum_{\pi \in \prod} T_\pi(t) = t,\ t \geq 0 \qquad (13)$$

For a function f, $\dot{f}(t)$ denotes its derivative at t, if one exists. Assuming $T_\pi(t)$ is differentiable and $\exists\ \pi'$ such that: $W^\pi(t) < W^{\pi'}(t)$, then MWM dictates that $\dot{T}_\pi(t) = 0$.

B. Rate Allocation under MWM

The service rate [13] $s_{ij}(t)$ for VOQ_{ij} is defined as follows:

$$s_{ij}(t) = \frac{1}{t} \sum_{\pi \in \prod} \pi_{ij} T_\pi(t)$$

It has been shown [3] that under admissible traffic for MWM, all queue-sizes $Q_{ij}(t) = 0, \forall t \geq 0$. From equations (11) – (13) for admissible traffic, we can infer that $s_{ij}(t) \geq \lambda_{ij},\ \ \forall t \geq 0$. We now characterize the rates $s(t) = [s_{ij}(t)]$ under inadmissible traffic using the notion of a *cycle*.

Definition 4 (Cycle) In an $N \times N$ bipartite graph, a cycle of length l is an ordered set of input-output pairs $\{(i_1, o_1), (i_2, o_2), \cdots, (i_l, o_l)\}$ comprised of edges connecting these pairs.

Theorem 3: Let $\Lambda = [\lambda_{ij}]$ s.t. $\lambda_{ij} > 0, \forall i, j$. Let $s = [s_{ij}]$ represent the rates allocated by MWM. Create an $N \times N$ bipartite graph with edge weights $w_{ij} = (\lambda_{ij} - s_{ij})^+$. For any cycle C_ℓ, if $s_{ij} > 0$ for all edges in C_ℓ, the following holds [14]:

$$\sum_{j=1}^{\ell} w_{i_j, o_j} = w_{i_1, o_l} + \sum_{j=1}^{\ell-1} w_{i_{j+1}, o_j} \qquad (14)$$

C. MWM vs. Max-Min Fairness

Theorem 3 does not give a simple closed-form rate allocation as in Theorem 1. However, we can use this result to show that MWM's rate allocation is not fair. For example, consider a 2×2 switch with the following arrival and permutation matrices:

$$\Lambda = \begin{bmatrix} 0.8 & 0.1 \\ 0.3 & 0.5 \end{bmatrix} \quad \pi_1 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \quad \pi_2 = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$$

Suppose that MWM schedules π_1 for $\alpha \in [0, 1]$ fraction of the time and π_2 for the remaining $(1-\alpha)$ fraction of the time. From Theorem 3, it follows that:

$$(0.8 - \alpha)^+ + (0.5 - \alpha)^+ = (0.3 - (1 - \alpha))^+ + (0.1 - (1 - \alpha))^+ \qquad (15)$$

MWM's departure rates and the Max-Min fair rate allocation below show that MWM lacks fairness.

$$R_{mwm} = \begin{bmatrix} 0.75 & 0.1 \\ 0.25 & 0.5 \end{bmatrix} \quad R_{mmf} = \begin{bmatrix} 0.7 & 0.1 \\ 0.3 & 0.5 \end{bmatrix}$$

VI. Fair-MWM

Analogous to Fair-LQF, we now present Fair-MWM, an algorithm that aims to incorporate fairness into MWM to ensure that at every output of the $N \times N$ switch, the rate allocation is Max-Min fair.

[13] The service rate may be higher than the arrival rate, as we assume that MWM schedules a complete matching every time.

[14] The proof can be found at http://simula.stanford.edu/∼neha/FMWM.pdf.

A. The Algorithm

Fair-MWM combines the advantages of MWM and Max-Min fairness to provide fair treatment to all $VOQs$, ensuring a small delay for uncongested queues. While all well-behaved [15] flows receive desired service, the offending flows are allotted an even fraction of the leftover bandwidth.

When a queue exceeds a pre-specified threshold, say a percentage of the buffer size [16], it is considered congested. Once a congested queue VOQ_{ij} is served, it gets blocked [17] for n_j time slots, where n_j is the number of non-empty queues containing packets headed for output j. In the scenario that multiple outputs have congested queues, n_j may be different for every oversubscribed output j. Once the blocking is accounted for, the matching is determined by MWM as before. The pseudocode for Fair-MWM is as follows:

```
//Information Collection
if (voq_size(voq_ID) >= threshold)
    add_to_congested_list(voq_ID);

// Scheduling
// Step1: MWM
MWM_schedule_unblocked_voqs();

//Step2: Blocking information
for (i=0; i<N; i++) {
  for (j=0; j<N; j++) {
   if (voq[i][j] is matched and congested)
      cycles_to_block[i][j] =
     number-of-nonempty-voqs-to-output-j();

   else if (cycles_to_block[i][j] > 0)
      cycles_to_block[i][j]--;
  }
}
```

B. Performance Evaluation

In this section, we study the rates allocated by the MWM and Fair-MWM algorithms by simulating various inadmissible traffic scenarios. The results show that Fair-MWM is indeed fair. We verified, in addition, that when traffic is admissible, the rate allocation is identical to that of MWM.

We consider a 4×4 example of an overloaded switch. Time is slotted and arriving traffic is Bernoulli IID. The performance of each algorithm is evaluated under various overloaded arrival traffic patterns (with multiple overloaded outputs). Simulations are run long enough to obtain the equilibrium results.

We first compare the performance of MWM and Fair-MWM with Max-Min fairness, when arriving traffic causes one output port to be overloaded. Consider the following arrival matrix:

$$\Lambda(1) = \begin{bmatrix} 1.0 & 0.0 & 0.0 & 0.0 \\ 0.15 & 0.2 & 0.2 & 0.2 \\ 0.15 & 0.2 & 0.2 & 0.2 \\ 0.15 & 0.2 & 0.2 & 0.2 \end{bmatrix}$$

The Max-Min fair and MWM rate allocations for $\Lambda(1)$ are:

$$R_{mmf} = \begin{bmatrix} 0.55 & 0.0 & 0.0 & 0.0 \\ 0.15 & 0.2 & 0.2 & 0.2 \\ 0.15 & 0.2 & 0.2 & 0.2 \\ 0.15 & 0.2 & 0.2 & 0.2 \end{bmatrix} \quad R_{mwm} = \begin{bmatrix} 0.88 & 0.0 & 0.0 & 0.0 \\ 0.04 & 0.2 & 0.2 & 0.2 \\ 0.04 & 0.2 & 0.2 & 0.2 \\ 0.04 & 0.2 & 0.2 & 0.2 \end{bmatrix}$$

[15] By well-behaved flows, we mean flows that ask for less than their fair share of service.

[16] The threshold may be chosen freely and does not affect the long-term performance of the algorithm.

[17] Blocking VOQ_{ij} is equivalent to assigning it a weight of 0.

Compared to the rate of 0.55 in $R_{mmf}(1)$, MWM allocates a service rate of 0.88 to the greedy queue VOQ_{11} and only 0.04 to the remaining VOQs ($VOQ_{21,31,41}$) destined for output 1. The Fair-MWM algorithm gives a rate allocation identical to that given by R_{mmf}.

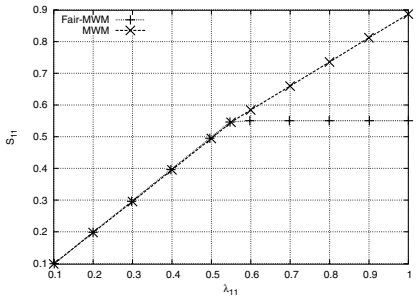

Fig. 2. MWM & Fair-MWM: Comparison of Rate Allocation

Instead of fixing λ_{11} at 1.0, we now vary it from 0.1 to 1.0 and evaluate the performance of Fair-MWM. Figure 2 depicts that Fair-MWM successfully limits the service rate of VOQ_{11} to 0.55, once λ_{11} exceeds the fair share. Under MWM, however, the service rate for VOQ_{11} grows monotonically as λ_{11} increases. Since the service rates for other queues destined for the same output are not protected, they become much smaller than the demand rates.

When there exist multiple offending flows destined for the same output, Fair-MWM first satisfies the service requirement for the good flows to that output and then splits the remaining service capacity evenly among the offenders. For instance, view the following example:

$$\Lambda(2) = \begin{bmatrix} 0.5 & 0.1 & 0.1 & 0.1 \\ 0.5 & 0.1 & 0.1 & 0.1 \\ 0.1 & 0.1 & 0.1 & 0.1 \\ 0.1 & 0.1 & 0.1 & 0.1 \end{bmatrix}$$

The Max-Min fair and MWM rate allocations for $\Lambda(2)$ are:

$$R_{mmf} = \begin{bmatrix} 0.4 & 0.1 & 0.1 & 0.1 \\ 0.4 & 0.1 & 0.1 & 0.1 \\ 0.1 & 0.1 & 0.1 & 0.1 \\ 0.1 & 0.1 & 0.1 & 0.1 \end{bmatrix} R_{mwm} = \begin{bmatrix} 0.45 & 0.1 & 0.1 & 0.1 \\ 0.45 & 0.1 & 0.1 & 0.1 \\ 0.05 & 0.1 & 0.1 & 0.1 \\ 0.05 & 0.1 & 0.1 & 0.1 \end{bmatrix}$$

The Fair-MWM and Max-Min fair rate allocations are identical.

When there exist offending flows destined for multiple outputs, Fair-MWM limits their service rates independently while protecting other non-offending flows. For example, consider the arrival matrix $\Lambda(3)$:

$$\Lambda(3) = \begin{bmatrix} 0.8 & 0.0 & 0.0 & 0.0 \\ 0.3 & 0.3 & 0.1 & 0.1 \\ 0.0 & 0.8 & 0.0 & 0.0 \\ 0.2 & 0.2 & 0.2 & 0.2 \end{bmatrix}$$

The MWM and Max-Min fair rate allocations for $\Lambda(3)$ are:

$$R_{mwm} = \begin{bmatrix} 0.7 & 0.0 & 0.0 & 0.0 \\ 0.2 & 0.2 & 0.1 & 0.1 \\ 0.0 & 0.7 & 0.0 & 0.0 \\ 0.1 & 0.1 & 0.2 & 0.2 \end{bmatrix} R_{mmf} = \begin{bmatrix} 0.5 & 0.0 & 0.0 & 0.0 \\ 0.3 & 0.3 & 0.1 & 0.1 \\ 0.0 & 0.5 & 0.0 & 0.0 \\ 0.2 & 0.3 & 0.2 & 0.2 \end{bmatrix}$$

This behavior of MWM is analogous to what we observed previously: greedy flows are served far more than their fair share. Fair-MWM solves this problem by allocating Max-Min fair rates: $R_{f-mwm} = R_{mmf}$.

We now test the robustness of Fair-MWM in a more complicated scenario, when there are multiple offending flows for multiple outputs. The arrival matrix is:

$$\Lambda(4) = \begin{bmatrix} 0.6 & 0.0 & 0.2 & 0.1 \\ 0.6 & 0.2 & 0.0 & 0.1 \\ 0.0 & 0.6 & 0.0 & 0.1 \\ 0.2 & 0.6 & 0.0 & 0.1 \end{bmatrix}$$

The MWM and Max-Min fair rate allocations for $\Lambda(4)$ are:

$$R_{mwm} = \begin{bmatrix} 0.47 & 0.0 & 0.2 & 0.1 \\ 0.47 & 0.06 & 0.0 & 0.1 \\ 0.0 & 0.47 & 0.0 & 0.1 \\ 0.06 & 0.47 & 0.0 & 0.1 \end{bmatrix} R_{mmf} = \begin{bmatrix} 0.4 & 0.0 & 0.2 & 0.1 \\ 0.4 & 0.2 & 0.0 & 0.1 \\ 0.0 & 0.4 & 0.0 & 0.1 \\ 0.2 & 0.4 & 0.0 & 0.1 \end{bmatrix}$$

Again, our algorithm performs a rate allocation that is Max-Min fair, i.e. $R_{f-mwm} = R_{mmf}$.

This concludes our study of fairness in switch-scheduling as we complete the analysis of Fair-MWM, having demonstrated the benefits it adds to the MWM scheduling algorithm.

VII. Conclusions

In practice, traffic is often inadmissible and queues are often overloaded. The inadmissible traffic scenario, however, has thus far been neglected. In this paper, we showed that the LQF and MWM algorithms considered ideal for admissible traffic perform poorly when traffic is inadmissible. Their allocation of service rates to input queues is biased in favor of offending (overloaded) traffic flows, penalizing the well-behaved flows unjustly. We addressed this problem by proposing algorithms based on the notion of Max-Min fairness, also highlighting the duality shared by Max-Min fairness and LQF. Finally, we observed that our algorithms, Fair-LQF and Fair-MWM, effectively perform a fair rate allocation with low delay, using the knowledge of queue sizes alone.

The techniques we employed in this paper are of a general nature and can easily be adapted to study the performance of other scheduling algorithms under inadmissible traffic.

References

[1] D. Bertsekas, R. Gallager, "Data Networks", *Prentice Hall*, 1992, pp. 526.

[2] J.G. Dai, "Stability of fluid and stochastic networks", Miscellanea Publication, No. 9, Center for Mathematical Physics and Stochastics, Denmark, `http://www.maphysto.dk`, Jan. 1999.

[3] J.G. Dai, B. Prabhakar, "The Throughput of Data Switches with and without Speedup" in *Proceedings of IEEE INFOCOM*, Mar. 2000, pp. 556-564.

[4] P. Giaccone, B. Prabhakar, D. Shah, "Switching under Energy Constraints" in *Asilomar Conference on Signals, Systems and Computers*, Nov. 2002.

[5] N. McKeown, A. Mekkittikul, V. Anantharam, J. Walrand, "Achieving 100% throughput in an input-queued switch" in *IEEE Transactions on Communications*, vol. 47, n. 8, Aug. 1999, pp. 1260-1267.

[6] D. Shah, D. Wischik, "Switch under Heavy Traffic", under preparation, 2004.

[7] L. Tassiulas, "Linear Complexity Algorithms for Maximum Throughput in Radio Networks and Input-Queued Switches" in *Proceedings of IEEE INFOCOM*, Apr. 1998, pp. 533-539.