Rate Stability of Stable-Marriage Scheduling Algorithms in Input-Queued Switches

Sunil Kumar
Graduate School of Business
Stanford University

Paolo Giaccone and Emilio Leonardi
Dipartimento di Elettronica
Politecnico di Torino

Abstract

Input queued switches are currently employed in high speed networks, since their architecture scales well with the link rate and the number of ports. However, they require a scheduling algorithm to select the packets to be transmitted across the switching fabric. Stable marriage algorithms provide an appealing solution for the scheduling problem, since they can be computationally efficient. In this paper we address the issue of rate stability of such algorithms using fluid model techniques. We derive the fluid model under stable marriage algorithms and show that the fluid model admits unstable trajectories. We provide a sufficient condition on the arrival rates for 2-diagonal traffic that ensures that any stable marriage algorithm is stable.

1 Introduction

Fast packet switches are ubiquitous in modern telecommunication networks. They receive incoming packets from input ports and forward them to output ports, according to some routing decision. An internal switching *fabric* physically transfers the packets from the input ports to the the output ports, with the most common example of such a fabric being the crossbar. To be efficiently implementable in hardware, the switch works with fixed size packets (called "cells") and runs in a synchronous fashion; if variable size packets are received (like in an IP router) they are segmented at the switch input ports into multiple cells, which are independently transferred to the output port, and then reassembled before transmission on the output link.

Depending on the internal architecture, switches can be divided mainly in two categories: Output-Queued (OQ) switches and Input-Queued (IQ) switches. In OQ switches, cells are immediately transferred on arrival to the respective outputs where they queue up for eventual departure on the output lines. In an OQ switch with N input lines and N output lines (usually termed an $N \times N$ switch), the switching fabric and the output memory must run at N times the input data rate. In this case we say that the switch exhibits a speed-up s equal to N.

IQ switches, on the other hand, queue up cells on arrival and transfer them to output lines when feasible: no more than one cell may be transferred from each input and to each output in each time slot. Indeed, both the switching fabric and output memories hardware need run only at a speed equal to the input data rate (i.e., $s = 1$). For this reason IQ switches appear particularly well suited for high-speed networks (see [15] for a discussion of the relative merits of IQ switches). This reduction in bandwidth requirement comes at a price, because it is necessary to design scheduling algorithms for IQ switches that choose which cells, among all cells queued at the inputs, will be transferred to outputs in a given time slot. The choice of the scheduling algorithm can have substantial impact on the performance of the IQ switch. For example, IQ switches which operate under a simple First-In First-Out (FIFO) scheduling rule at input level can suffer from so-called Head of Line (HOL) blocking [12]. When HOL blocking occurs, the sustainable throughput of the switch is reduced. For i.i.d. Bernoulli traffic, uniformly distributed among all the inputs and the outputs, the achieved throughput is reduced by HOL blocking to 58% of the maximum sustainable throughput [12]. HoL blocking can be avoided by adopting a Virtual Output Queue (VOQ) scheme, in which one queue is present for each input-output couple.

A variety of scheduling algorithms have been shown to achieve 100% throughput in an IQ switch with VOQ, as long as no input or output line is loaded beyond its capacity, cf. [6, 10, 15, 16, 20]. We refer to such algorithms as being *rate stable* in the rest of this paper. Most of these algorithms suffer from the following drawback. In order to determine which cells which be transferred in a given slot, the controller has to perform a substantial amount of computation.

A class of scheduling algorithms that is both simple to describe and computationally parsimonious is the class of so-called stable marriage (SM) algorithms. The defining feature of such an algorithm is the following. Consider a crossbar switch with speed up $s = 1$. In any time slot, a cell is not transferred from input i to output j only if a cell is being transferred from either another input i' to output j or from input i to another output j', and the number of cells waiting to go from i' to j or from i to j' is no smaller than the number of cells waiting to go from i to j. Thinking of inputs as brides, outputs as grooms, connections from inputs to outputs as marriages, and the number of cells waiting to go from a specified input to a specified output as the relative preferability of the corresponding marriage, an SM algorithm never marries bride i to groom j' and bride i' to groom j when i and j would rather be married to each other. An example of an SM algorithm is the so-called Greedy Longest Queue First (GLQF) algorithm [11], which first connects the input-output pair with the maximal queue, then the pair with the maximal queue among all those that can be feasibly connected, and so on. An SM scheduling algorithm can be implemented using the matching algorithm due to Gale and Shapley [9]. In this algorithm, an unmatched groom i proposes to the most preferred bride j that he has not proposed to so far, and bride j accepts the proposal if either she is unmatched or if she prefers i over her current match. This procedure is repeated until there are no more unmatched grooms. This deterministic algorithm has complexity $O(N^2)$ and does not require additional memory.

In this paper we will analyze rate stability of SM algorithms in IQ switches. Existing work allows one to deduce that SM algorithms are rate stable if the total load on every input port and every output is less than 1/2 [6] or if ties are broken in an intelligent manner [18]. However, one cannot break ties intelligently without increasing the computational complexity of the SM algorithm. In this paper we assume that tie breaking need not take place in an intelligent manner and hence the results of [18] do not apply. We analyze SM algorithms using the fluid model approach pioneered by Dai and his co-workers [2, 3, 4, 5, 6] and first applied to input queued switches in [6]. This approach allows us to operate under very weak assumptions on the traffic arriving at the switch – we only need that it satisfy the law of large numbers. We derive the fluid model associated with SM algorithms and show that the fluid model admits unstable trajectories,

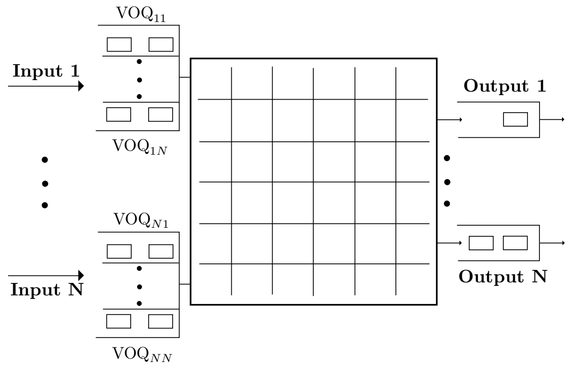

Figure 1: A Crossbar Switch, drawn after Dai and Prabhakar [6]

even though the obvious capacity constraints are satisfied by the arrival rates. The key to establishing this negative result is showing that ties can be persistent in the fluid model and that they can be broken constantly in a malicious way. This negative result also indicates that SM algorithms need not always be rate stable.

Thus, we are led to search for conditions on arrival rates, beyond the obvious capacity constraints, under which SM algorithms are guaranteed to be rate stable. A common traffic pattern used in the analysis of scheduling algorithms is the so-called diagonal (or more accurately, 2-diagonal) traffic [10] in which each input port i has traffic only for output ports i and $i+1$ modulo N. We denote the arrival rate from input port i to output port j as λ_{ij}. For the 2-diagonal traffic pattern, we show that if we have $\lambda_{ii} + \lambda_{i,i+1} + \lambda_{i+1,i+1} \leq 1$ or $\lambda_{i-1,i} + \lambda_{ii} + \lambda_{i,i+1} \leq 1$ for *just one* input port i, any SM algorithm is stable, regardless of how ties are broken. This is a fairly weak assumption, since we require only one port to satisfy this condition. But, this condition is neither necessary nor easily extendable to general traffic patterns. Finding better sufficient conditions for stability remains the focus of our future work.

2 Model

In this section, and in the rest of the paper, we will stick to the notation and framework of Dai and Prabhakar [6]. Consider an $N \times N$ crossbar switch such as the one shown in Figure 1, drawn after Dai and Prabhakar [6]. We work with the following discrete time framework. Time is slotted and cells arrive at the switch at the beginning of a time slot. For concreteness, time slot n corresponds to the time interval $[n-1, n)$, $n = 1, 2 \ldots$. Each input has an infinite storage buffer for holding cells prior to switching them to their respective outputs. As introduced before, we consider each buffer at an input as being partitioned into N "virtual output queues" (VOQ's), each of infinite storage capacity, with the convention that virtual output queue VOQ_{ij} holds cells arriving at input i destined for output j. We denote the contents of VOQ_{ij} at the beginning of time slot n by $Z_{ij}(n)$, including any arrivals at time $n-1$.

We assume that the speed up of the switch $s = 1$. That is, in each period the switch performs the following two operations. First, based on the scheduling algorithm employed, a matching between inputs and outputs is chosen in such a way that no input may be matched to more than one output and no output can be matched to more than one input. Second, if input i is matched to output j and the VOQ_{ij} is non-empty, then a cell is transferred from i to j where it departs from the switch. We assume that cells depart from the switch just prior to the end of a time slot.

A matching may be represented by a permutation matrix π. That is, input i is matched to output j if and only if $\pi_{ij} = 1$. Let Π be the set of all $N!$ permutation matrices. As can be seen from the description above, a scheduling algorithm is specified by the matchings that it chooses. To be more specific, for a permutation $\pi \in \Pi$, let $T_\pi(n)$ denote the cumulative amount of time that permutation π has been used by time slot n under the scheduling algorithm used, with the convention that $T_\pi(0) = 0$. Then, knowing a scheduling algorithm is equivalent to knowing $T_\pi(n)$ for all $\pi \in \Pi$, and for all $n = 1, 2, \ldots$.

Let $A_{ij}(n)$ denote the number of cells that have arrived at input i destined for output j up to time slot n. Since we assume that cell arrivals occur at the beginning of a time slot, for any time $t \in (n-1, n)$, $A_{ij}(n)$ is the cumulative number of cells that have arrived at VOQ_{ij} by time t. We adopt the convention that $A_{ij}(0) = 0$. We assume that the arrival processes $\{A_{ij}(\cdot), i, j = 1, \ldots, N\}$ satisfy a strong law of large numbers (SLLN): with probability one,

$$\lim_{n\to\infty} \frac{A_{ij}(n)}{n} = \lambda_{ij} \qquad i, j = 1, \ldots, N. \tag{1}$$

We call λ_{ij} the arrival rate at VOQ_{ij}. Assumption (1) on arrival processes is very mild. It is satisfied, for example, when the arrival processes $\{A_{ij}(\cdot), i, j = 1, \ldots, N\}$ are jointly stationary and ergodic with arrival rates λ_{ij}. Furthermore, we assume that no input or output is loaded beyond its capacity:

$$\sum_{i=1}^{N} \lambda_{ij} \le 1 \quad \text{for each } j = 1, 2, \ldots, N, \text{ and} \tag{2}$$

$$\sum_{j=1}^{N} \lambda_{ij} \le 1 \quad \text{for each } i = 1, 2, \ldots, N. \tag{3}$$

Let $D_{ij}(n)$ be the number of departures from $\text{VOQ}_{ij}(n)$ up to time slot n, with the convention that $D_{ij}(0) = 0$. Note that $D_{ij}(n) = D_{ij}(n-1)+1$ if and only if the scheduling algorithm chooses a matching π in time slot n such that $\pi_{ij} = 1$, and $Z_{ij}(n) > 0$.

Definition 1 *A switch operating under a scheduling algorithm is said to be rate stable if, with probability one,*

$$\lim_{n\to\infty} \frac{D_{ij}(n)}{n} = \lambda_{ij} \qquad i, j = 1, \ldots, N \tag{4}$$

for any arrival processes satisfying (1-3).

In other words, a switch operating under a rate stable scheduling algorithm can achieve up to 100% throughput, if there is enough offered load.

In Sections 4 and 5 we will consider a special traffic pattern called *2-diagonal traffic* defined below.

Definition 2 *A switch is said to operating in 2-diagonal traffic if $\lambda_{ij} = 0$ for $j \neq i$ and $j \neq (i+1) \bmod N$.*

3 Stable-Marriage Algorithms

In this section, we will define the SM algorithm, describe the dynamics of the crossbar switch with $s = 1$ operating under this algorithm and obtain its corresponding fluid model.

From the model description in Section 2, we conclude that the switch dynamics, i.e., the equations satisfied by the various quantities defined in Section 2, under any scheduling policy, are as follows.

$$Z_{ij}(n) = Z_{ij}(0) + A_{ij}(n) - D_{ij}(n), \tag{5}$$

$$D_{ij}(n) - D_{ij}(n-1) = \sum_{\pi \in \Pi} \pi_{ij} 1_{\{Z_{ij}(n)>0\}} (T_\pi(n) - T_\pi(n-1)), \tag{6}$$

$$T_\pi(\cdot) \text{ is non decreasing and } \sum_{\pi \in \Pi} T_\pi(n) = n. \tag{7}$$

We augment the equations (5-7) with an equation that is specific to SM algorithms. Consider the situation when a buffer VOQ_{ij} is not being matched at time n under a stable marriage algorithm for some $i, j = 1, 2, \ldots, N$, with $Z_{ij}(n) > 0$. This can happen only if there exists either a $k \neq i$ such that $Z_{kj} \geq Z_{ij}$ and input k is matched to output j or an $\ell \neq j$ such that $Z_{i\ell} \geq Z_{ij}$ and input i is matched to output ℓ. If no such k or ℓ existed, input i would strictly prefer output j over the current match ℓ and output j would strictly prefer input i over the current match k and hence the current matching would not result in a stable marriage. Thus, under any SM algorithm, for any $i, j = 1, 2, \ldots, N$, we must have[1]:

$$\sum_{\pi \in \Pi} g_{ij}(Z(n), \pi)(T_\pi(n) - T_\pi(n-1)) = 0, \text{ for each } i, j, \text{ where} \tag{8}$$

$$g_{ij}(Z, \pi) = \sum_{k=1}^{N} \pi_{kj}(Z_{ij} - Z_{kj})^+ \bigwedge \sum_{\ell=1}^{N} \pi_{i\ell}(Z_{ij} - Z_{i\ell})^+ \bigwedge 1. \tag{9}$$

To derive Eq.(9), observe that $g_{ij}(Z, \pi) > 0$ if there exist $k \neq i$ and $l \neq j$ such that: $\pi_{kj} = 1$, $\pi_{il} = 1$, $Z_{ij} > Z_{kj}$ and $Z_{ij} > Z_{il}$. This occur only if VOQ_{ij}, which does not belong to the matching, has more cells than both queues VOQ_{kj} and VOQ_{il}, which belong to the matching; this means that the matching is not stable.

Equation (8,9) will be the only feature of SM algorithms that we will use in the rest of the paper, and hence we define a stable marriage algorithm as follows.

Definition 3 *Any algorithm under which the dynamics of the crossbar switch with speed up $s = 1$ satisfies (5-9) is defined to be a stable marriage (SM) algorithm.*

We will analyze the rate stability of the algorithm defined above using the fluid model approach pioneered by Dai [2, 3, 4, 5, 7], and first applied to scheduling algorithms in IQ switches in [6]. The approach can be described as consisting of the following steps.

1. Write down continuous time analogs $(Z(t), D(t), T(t))$ of the discrete time processes $(Z(n), T(n), D(n))$, and scale them using a law of large numbers scaling, $(Z^r(t), D^r(t), T^r(t)) = \frac{1}{r}(Z(rt), D(rt), T(rt))$.

2. Show that $(Z^r(t), D^r(t), T^r(t))$ have limits as $r \to \infty$ along subsequences, hereafter referred to as fluid limits.

3. Find a set of equations that every fluid limit must satisfy, hereafter referred to as the fluid model equations, and show that every solution to the fluid model equations must have the property that if $Z(0) = 0$, then $Z(t) = 0$ for all $t > 0$.

[1] As usual, we use x^+ to denote $\max(x, 0)$ and $x \wedge y$ to denote $\min(x, y)$.

4. Translate this property of fluid model solutions to rate stability of the original discrete time stochastic system.

Recall that $Z_{ij}(n)$ is the number of cells on VOQ_{ij} at the beginning of time slot n. We extend the definition of $Z_{ij}(t)$, for arbitrary time $t \geq 0$, to be $Z_{ij}(\lfloor t \rfloor)$, where $\lfloor t \rfloor$ is the largest integer less than or equal to t. Then $Z_{ij}(\cdot) \in \mathcal{D}[0,\infty)$, where, for an integer d, $\mathcal{D}^d[0,\infty)$ is the space of functions $f : [0,\infty) \to \mathcal{R}^d$ that are right continuous and have left limits in $(0,\infty)$. Similarly, we can extend the definition of $A(t)$ so that it is defined for $t \geq 0$. Note that the functions $A_{ij}(\cdot)$ and $Z_{ij}(\cdot)$ are elements of $\mathcal{D}[0,\infty)$, in general. We define $D(t)$ and $T_\pi(t)$ for $t \geq 0$ so that they are *continuous* functions using the following piecewise linear interpolation. For $t \in (n, n+1)$, let $D(t) = D(n)+(t-n)(D(n+1)-D(n))$ and let $T_\pi(t) = T_\pi(n) + (t-n)(T_\pi(n+1) - T_\pi(n))$. Note that the functions $D_{ij}(t)$ and $T_\pi(t)$ are random elements of $\mathcal{C}[0,\infty)$. We use the notation $A(\cdot,\omega)$, $D(\cdot,\omega)$ $T_\pi(\cdot,\omega)$ and $Z(\cdot,\omega)$ to explicitly denote the dependence on the ω, a sample point in our reference probability space $(\Omega, \mathcal{F}, P)$. Finally, for any absolutely continuous function f, we use the notation $\dot{f}$ to denote df/dt whenever the derivative exists. Now, for each $r > 0$ define

$$\bar{A}^r(t,\omega) = r^{-1}A(rt,\omega), \qquad \bar{D}^r(t,\omega) = r^{-1}D(rt,\omega), \tag{10}$$

$$\bar{T}^r(t,\omega) = r^{-1}T(rt,\omega), \qquad \bar{Z}^r(t,\omega) = r^{-1}Z(rt,\omega). \tag{11}$$

Lemma 1 *For each ω satisfying (1) and any sequence $\{r\}$ with $r \to \infty$, there exists a subsequence $\{r_n\}$ and a Lipschitz function $(\bar{D}(\cdot,\omega), \bar{T}(\cdot,\omega), \bar{Z}(\cdot,\omega))$, hereafter called a* fluid limit, *such that*

$$\left(\bar{D}^{r_n}(\cdot,\omega), \bar{T}^{r_n}(\cdot,\omega), \bar{Z}^{r_n}(\cdot,\omega)\right) \to \left(\bar{D}(\cdot,\omega), \bar{T}(\cdot,\omega), \bar{Z}(\cdot,\omega)\right) \tag{12}$$

uniformly on compact time sets as $n \to \infty$. Furthermore, each fluid limit satisfies the following set of equations, hereafter referred to as fluid model equations *at almost every time t, that is, whenever the derivatives exist*[2].

$$Z_{ij}(t) = Z_{ij}(0) + \lambda_{ij}t - D_{ij}(t) \geq 0, \qquad t \geq 0,\ i,j = 1,\ldots,N, \tag{13}$$

$$\dot{D}_{ij}(t) = \sum_{\pi \in \Pi} \pi_{ij}\dot{T}_\pi(t), \qquad \text{if } Z_{ij}(t) > 0, \qquad i,j = 1,\ldots,N, \tag{14}$$

$$T_\pi(\cdot) \text{ is non}-\text{decreasing}, \quad \sum_{\pi \in \Pi} T_\pi(t) = t, \qquad t \geq 0, \text{ and} \tag{15}$$

$$\sum_{\pi \in \Pi} g_{ij}(Z(t),\pi)\dot{T}_\pi(t) = 0,\ i,j = 1,\ldots,N, \tag{16}$$

where, $g_{ij}(.,.)$ is given by (9).

Proof. The convergence along subsequences and the consequent existence of Lipschitz fluid limits is proved in the appendix of Dai and Prabhakar [6]. A proof that each fluid limit satisfies (13-15) is also provided there, and we will not repeat the arguments in the interest of brevity. We will only provide proof that every fluid limit satisfies (16), which is particular to our choice of SM scheduling algorithms. Equation (8) combined with the continuous extensions of (Z^r, D^r, T^r) yields

$$\int_0^t g_{ij}(Z^r(s),\pi)dT^r_\pi(s) = 0,$$

[2] We abuse notation slightly and use the same symbols for both discrete quantities and their counterparts in the fluid model equations. The usage is clear from the context.

for each $t > 0$, $i, j = 1, \ldots, N$ and permutation matrix $\pi \in \Pi$. For each $i, j = 1, \ldots, N$ and $\pi \in \Pi$, $g_{ij}(., \pi)$ is a continuous bounded function and $T_\pi(.)$ is a non-decreasing function. By Lemma 2.4 of Dai and Williams [8], we have

$$0 = \int_0^t g_{ij}(Z^{r_n}(s), \pi) dT_\pi^{r_n}(s) \rightarrow \int_0^t g_{ij}(Z(s), \pi) dT_\pi(s),$$

for each $t > 0$, $i, j = 1, \ldots, N$ and permutation matrix $\pi \in \Pi$, from which (16) follows. □

Theorem 1 (Dai and Prabhakar) *A switch is rate stable if the corresponding fluid model has the following property. Every solution (D, T, Z) to the fluid model equations (13-16) satisfies*

$$Z(0) = 0 \Rightarrow Z(t) = 0 \textit{ for all } t \geq 0. \tag{17}$$

Proof. Although the arguments are identical to the proof of Theorem 3 in Dai and Prabhakar [6], we reproduce the arguments because their statement did not include (16). For each ω satisfying (1), from Lemma 1 and the fact that $Z_{ij}(n)$ is finite when $n = 0$ in the discrete system, the fluid limit $(\bar{Z}, \bar{D}, \bar{T})$ satisfies $\bar{Z}_{ij}(0) = 0$. Hence, from (17) the fluid limit satisfies $\bar{Z}_{ij}(t) = 0$ for $t \geq 0$. Using this in (13), we get that $\bar{D}_{ij}(t) = \lambda_{ij} t$ for $t \geq 0$. Thus, from Lemma 1 and a standard subsequence argument, $\bar{D}_{ij}^r(1, \omega) \rightarrow \lambda_{ij}$ as $r \rightarrow \infty$ or $\lim_{r \rightarrow \infty} D_{ij}(r, \omega)/r = \lambda_{ij}$. Restricting r to the integers on the left-hand side yields (4), thus proving the theorem. □

4 Unstable Fluid Trajectories under 2-diagonal Traffic

Recall definition 2 of 2-diagonal traffic. Consider a 3×3 switch operating under 2-diagonal traffic with arrival rates given by $\lambda_{11} = \lambda_{22} = \lambda_{33} = \lambda_{12} = \lambda_{23} = \lambda_{31} = 2/5$. All other λ_{ij} are zero. Note that these arrival rates satisfy (2,3). Consider three permutation matrices given by

$$\pi^1 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix}, \ \pi^2 = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix} \text{ and } \pi^3 = \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}.$$

Consider the processes: $T_{\pi^1}(t) = T_{\pi^2}(t) = T_{\pi^3}(t) = \left(\frac{1}{3}\right) t$; $D_{ij}(t) = \sum_{k=1}^3 \pi_{ij}^k T_{\pi^k}(t)$; then $Z_{ii}(t) = Z_{i,i+1}(t) = \left(\frac{2}{5} - \frac{1}{3}\right) t$ for $i = 1, 2, \ldots, N-1$ and $Z_{NN}(t) = Z_{N1}(t) = \left(\frac{2}{5} - \frac{1}{3}\right) t$ for all $t \geq 0$. The reader can verify the processes above satisfy (13-16) with $Z_{ij}(0) = 0$ for all i, j. Yet Z_{ii} and $Z_{i,i+1}$ grow linearly in time and hence (17) is not satisfied.

This example establishes that (17) need not always hold for switches operating under stable marriage algorithms. Hence it is not always possible to use the fluid model machinery to establish rate stability. Furthermore, it suggests that switches operating under stable marriage algorithms may be unstable. This is indeed the case [13].

5 Rate Stability under 2-diagonal Traffic

It is clear from the example in the previous section that additional assumptions are required if one is to establish rate stability under 2-diagonal traffic using Theorem 1. Theorem 2 below gives one such assumption on the arrival rates.

Theorem 2 *An input queued switch with speedup $s = 1$, operating under 2-diagonal traffic is rate stable under a stable marriage algorithm if there exists an $i \in \{1, 2, \ldots, N\}$ such that either*

$$\lambda_{ii} + \lambda_{i,i+1} + \lambda_{i+1,i+1} \leq 1 \qquad or \tag{18}$$

$$\lambda_{i-1,i} + \lambda_{ii} + \lambda_{i,i+1} \leq 1, \tag{19}$$

where the indices are interpreted as being modulo N.

The sufficient conditions (18,19) are weaker than assuming that the speedup $s = 2$. That is, (18,19) allow the load on inputs and outputs to be arbitrarily close to unity. Hence, Theorem 2 does not follow from the results of Dai and Prabhakar [6]. We will prove Theorem 2 by showing that any solution (D, T, Z) to the fluid model equations (13-16) satisfies (17) when (18,19) hold. Under 2-diagonal traffic, when we verify (17), we can ignore all Z_{ij} such that $j \neq i$ and $j \neq (i+1)$ modulo N. These buffers will never leave zero since their $\lambda_{ij} = 0$. Let $Z^*(t) = \max_{i,j} Z_{ij}(t)$ and let $M(t) = \{(i,j) \mid Z_{ij}(t) = Z^*(t)\}$ be the set of maximizers at time t. Note that $Z_{ij}(.)$ are all absolutely continuous functions of time, $Z^*(.)$ is also an absolutely continuous function of time. Therefore $\dot{Z}^*(t)$ exists at almost all times t, hereafter called regular times. A property of the maximum function, proved in [7] Lemma 3.2, yields:

Lemma 2 *At all regular times t, $\dot{Z}^*(t) = \dot{Z}_{ij}(t)$* for all *$(i,j) \in M(t)$.*

In this section, we will always interpret all indices as modulo N. The next lemma establishes a crucial property that renders the proof of Theorem 2 essentially trivial.

Lemma 3 *At a regular time t, suppose that $Z^*(t) > 0$ and $\dot{Z}^*(t) > 0$. If, for some i, $(i,i) \in M(t)$, then*

$$(i, i+1) \in M(t) \text{ and } (i-1, i) \in M(t), \text{ and} \tag{20}$$

$$\lambda_{ii} + \lambda_{i,i+1} + \lambda_{i-1,i} > 1. \tag{21}$$

Similarly, if for some i, $(i, i+1) \in M(t)$, then

$$(i, i) \in M(t) \text{ and } (i+1, i+1) \in M(t), \text{ and} \tag{22}$$

$$\lambda_{ii} + \lambda_{i,i+1} + \lambda_{i+1,i+1} > 1. \tag{23}$$

Proof. At a regular time t, suppose $(i,i) \in M(t)$. From Lemma 2, $\dot{Z}_{ii}(t) > 0$. From (16),

$$\sum_{\pi \in \Pi} g_{ii}(Z, \pi) \dot{T}_\pi = 0, \tag{24}$$

where $g_{ii}(Z, \pi)$ is given by (9). Suppose $(i, i+1) \notin M(t)$ *and* $(i-1, i) \notin M(t)$. This implies that if $\dot{T}_\pi(t) > 0$ for some $\pi \in \Pi$, then $\pi_{ii} = 1$ from (24). This would mean that $\dot{Z}_{ii}(t) = \lambda_{ii} - 1$ from (13-15). This contradicts $\dot{Z}_{ii}(t) > 0$ from the assumptions in (2-3). So either $(i, i+1) \in M(t)$ or $(i-1, i) \in M(t)$ (or both). Suppose $(i-1, i) \in M(t)$ but $(i, i+1) \notin M(t)$. In this case, if $\dot{T}_\pi(t) > 0$ for some $\pi \in \Pi$, then $\pi_{ii} + \pi_{i-1,i} = 1$ from (24). From (15) we have

$$\sum_{\pi \in \Pi} (\pi_{i-1,i} + \pi_{ii}) \dot{T}_\pi(t) = 1.$$

From Lemma 2, $\dot{Z}_{i-1,i}(t) = \dot{Z}_{ii}(t) > 0$. Thus, from (13,14)

$$0 < \dot{Z}_{i-1,i}(t) + \dot{Z}_{ii}(t) = \lambda_{i-1,i} + \lambda_{ii} - \sum_{\pi\in\Pi}(\pi_{i-1,i} + \pi_{ii})\dot{T}_\pi(t) = \lambda_{i-1,i} + \lambda_{ii} - 1,$$

which contradicts (3). Hence, both $(i, i+1) \in M(t)$ and $(i-1, i) \in M(t)$, establishing (20). Now, from Lemma 2 and (13,14), we have

$$0 < \dot{Z}_{i-1,i}(t) + \dot{Z}_{ii}(t) + \dot{Z}_{i,i+1}(t) = \lambda_{i-1,i} + \lambda_{ii} + \lambda_{i,i+1} - \sum_{\pi\in\Pi}(\pi_{i-1,i} + \pi_{ii} + \pi_{i,i+1})\dot{T}_\pi(t).$$

Hence, to establish (21) it suffices to show that

$$\sum_{\pi\in\Pi}(\pi_{i-1,i} + \pi_{ii} + \pi_{i,i+1})\dot{T}_\pi(t) \geq 1. \tag{25}$$

From (24), if $\dot{T}_\pi(t) > 0$, then $\pi_{i-1,i} + \pi_{ii} + \pi_{i,i+1} \geq 1$. That is, if a permutation matrix π is picked by a stable marriage algorithm when Z_{ii} is a maximizer then a maximizer is connected in either the i-th column or the i-th row by the permutation. This establishes (25) and hence (21)[3]. The proof of (22,23) is identical and omitted. □

Proof of Theorem 2. Suppose that (19) holds for some i. We will prove (17) when (19) holds for some i and omit the essentially identical proof for the case when (18) holds. The proof is based on constructing a Lyapunov function for Z, namely, $Z^*(t)$. If we can show that at all regular times when $Z^*(t) > 0$, we must have $\dot{Z}^*(t) \leq 0$, we are done because

$$(Z^*(t))^2 - (Z^*(0))^2 = 2\int_0^t Z^*(s)\dot{Z}^*(s)ds \leq 0.$$

Hence if $Z^*(0) = 0$, we have $Z^*(t) = 0$ for all $t \geq 0$, yielding (17).

Consider a regular time t when $Z^*(t) > 0$. Now assume (to establish the result by contradiction) that we have $\dot{Z}^*(t) > 0$. If $(i,i) \in M(t)$, then (21) must hold by Lemma 3. This is a contradiction to (19). Hence, $(i,i) \notin M(t)$. Now suppose that $(i, i+1) \in M(t)$. Then from Lemma 3 $(i,i) \in M(t)$ by (22). This is a contradiction to the result just established. Hence, $(i, i+1) \notin M(t)$. Now applying Lemma 3, for $(i+1, i+1)$ yields $(i+1, i+1) \notin M(t)$ by (20). If we continued this procedure we would end up with the absurd result that $M(t)$ is empty. Hence $\dot{Z}^*(t) > 0$ cannot hold when (19) holds. This establishes (17) as discussed in the previous paragraph. The case when (18) holds is essentially identical and is omitted. □

Acknowledgments. The authors would like to thank Balaji Prabhakar for bringing about this collaboration, and Frank Kelly, Mike Harrison, and Damon Wischik for useful discussions.

[3]We do not have equality in (25) because both $\pi_{i-1,i}$ and $\pi_{i,i+1}$ can be positive for a permutation π.

References

[1] P. Billingsley: *Convergence of Probability Measures.* John Wiley & Sons, New York, 1968.

[2] J.G. Dai: "On positive Harris recurrence of multiclass queuing networks: A unified approach via fluid limit models," *Annals of Applied Probability,* Vol. 5, pp. 49–99, 1995

[3] J. G. Dai: "A fluid-limit model criterion for instability of multiclass queuing networks," *Annals of Applied Probability*, Vol. 6, pp. 751–757, 1996.

[4] J. G. Dai: "Stability of fluid and stochastic processing networks," Miscellanea Publication, No. 9, Centre for Mathematical Physics and Stochastic, Denmark (http://www.maphysto.dk), Jan. 1999.

[5] J. G. Dai and S. P. Meyn: "Stability and convergence of moments for multiclass queuing networks via fluid limit models," *IEEE Transactions on Automatic Control,* Vol. 40, pp. 1889–1904, 1995.

[6] J. G. Dai and B. Prabhakar: "The throughput of data switches with and without speedup," *IEEE INFOCOM'00*, Vol. 2, pp. 556-564, Tel Aviv, Mar. 2000.

[7] J. G. Dai and G. Weiss: "Stability and instability of fluid models for reentrant lines," *Mathematics of Operations Research,* Vol. 21, pp. 115–134, 1996.

[8] J. G. Dai and R. J. Williams: " Existence and Uniqueness of Semimartingale Reflecting Brownian Motions in Convex Polyhedrons," *Theory of Probability and Its Applications,* Vol. 40, pp. 3–53, 1995.

[9] D. Gale and S. Shapley: "College Admissions and the Stability of Marriage," *American Mathematical Monthly,* Vol. 69, pp.9–15, 1962.

[10] P. Giaccone, B. Prabhakar and D. Shah: "Toward Simple High-performance Schedulers for High-Aggregate Bandwidth Switches," *IEEE INFOCOM'02*, Vol. 3, pp. 1160–1169, New York, June 2002.

[11] A.C. Kam, K.Y. Siu: "Linear-complexity algorithms for QoS support in input-queued switches with no speedup," *IEEE Journal on Selected Areas in Communications*, Vol. 17, No. 6, pp. 1040-56, 1999.

[12] M Karol, M. Hluchyj and S. Morgan: "Input Versus Output Queuing on a Space Division Switch," *IEEE Transactions on Communications,* Vol. 35, No. 12, pp. 1347–1356, 1987.

[13] E. Leonardi and P. Giaccone: "Further Comments on iLQF Stability," Personal Communication, 2002.

[14] N. McKeown: "iSLIP: A Scheduling Algorithm for Input-Queued Switches," *IEEE Transactions on Networking,* Vol. 7, pp. 188–201, Apr. 1999.

[15] N. McKeown, V. Anantharam, J. Walrand: "Achieving 100% Throughput in an Input-Queued Switch," *IEEE INFOCOM'96,* pp. 296–302, 1996.

[16] N. McKeown, A. Mekkittikul, V. Anantharam and J. Walrand: "Achieving 100% Throughput in an Input-Queued Switch," *IEEE Transactions on Communications,* Vol. 47, No. 8, Aug. 1999.

[17] B. Prabhakar and N. McKeown: "On the speedup required for combined input- and output-queued switching," *Automatica,* Vol. 12, No. 35, pp. 1909-1920, Dec. 1999.

[18] D. Shah: "Stable Algorithms for Input Queued Switches," *Proc. 39th Annual Allerton Conference on Communication, Control and Computing,* Oct. 2001.

[19] L. Tassiulas: "Linear complexity algorithms for maximum throughput in radio networks and input queued switches," *IEEE INFOCOM'98*, Vol. 2, pp.533-539, New York, 1998

[20] L. Tassiulas and A. Ephremides: "Stability properties of constrained queuing systems and scheduling policies for maximum throughput in multihop radio networks," *IEEE Transactions on Automatic Control,* Vol. 37, No. 12, pp. 1936–1948, 1992.