Distributed scheduling policies of low complexity for networks of input-queued switches

Claus Bauer

Dolby Laboratories, San Francisco, CA, 94103, USA,
cb@dolby.com

Abstract. Scheduling algorithms for input-queued switches have been widely researched. It has been shown that various classes of scheduling algorithms guarantee the stability of single switches. However, recent research has demonstrated that most of these scheduling algorithms do not the guarantee stability for networks of switches. Most of the research that treats networks of switches proposes switching policies that require coordination among switches. The problem to find *distributed* scheduling policies that guarantee the stability of a network of switches has so far only been investigated for a policy based on a computationally very complex maximum weight matching algorithm. In this paper, we investigate a class of *distributed* scheduling algorithms of *low complexity* that are based on maximal weight matching algorithms. We prove the stability of networks of input-queued switches where each switch deploys any maximal weight matching algorithm of the defined class.

1 Introduction and Motivation

The progress in optical transmission technologies creates a need for fast switching technologies in the internet core. Research on switch architectures and scheduling algorithms has mostly focused on input-queued (IQ) and combined input output-queued (CIOQ) switches. A typical CIOQ switch is shown in figure 1. To avoid head-of-line blocking, a typical NxN CIOQ switch has N virtual output queues $VOQ_{i,j}$, $1 \leq j \leq N$ at each input i. Packets that arrive at input i and are destined for output j and not forwarded immediately upon their arrival, are buffered in $VOQ_{i,j}$. The switching core works with a speedup of $S, S \geq 1$, i.e., it works at a speed S times faster than the speed of the input and the output links. If $S > 1$, packets are also buffered at the outputs. The switching core is typically modeled as a crossbar, such that not more than one packet can be sent simultaneously from the same input or to the same output.

The choice of the scheduling algorithm is a major design criteria for switches. The problem of finding an optimal switch configuration for a specific switch state can be modeled as the problem of finding a maximum weight matching of a bipartite $N \times N$ graph. In [8] and [9], it has been shown that for a speedup of $S = 1$, a maximum weight matching algorithm can guarantee the stability of a single switch if the weights are chosen proportionally to the lengths of the VOQs. However, the implementation of maximum weight matching algorithms

is impractical as they require the solution of an optimization problem that is based on the Hungarian method and has a known complexity of $O(N^3 \log N)$ (see [8]).

Therefore, the computationally less complex class of maximal weight match-

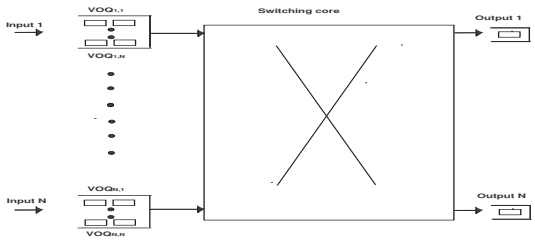

Fig. 1. Architecture of an input-queued switch

ing algorithms has been widely investigated ([7]). It has been shown that under the assumption of admissible traffic, every maximal weight matching algorithm deployed with a speedup of 2 ([6]) or even slightly less ([4],[5]) guarantees the stability of a single switch.

In [1] and [3], switching policies that guarantee the stability of all switches within a network, but require coordination and cooperation among the switches within the network, are presented. In [2], for the first time a *distributed* switching policy was proposed that guarantees 100% throughput in a network of input-queued switches. Each switch applies a maximum weight matching algorithm to configure its scheduling matrix independently of the other switches and no additional signaling traffic between the switches is required.

Due to the high computational complexity of maximum weight matching algorithms, it is of interest to understand if *distributed* scheduling algorithms of *low complexity* that guarantee 100% throughput in a network of input-queued switches exist. This paper shows for the first time the existence of *distributed* switching policies of *low complexity* that are based on maximal weight matching algorithms and that guarantee 100% throughput. We use the theory of the Lyapunov function ([8]) and the fluid model methodologies ([6]) to establish our results. Our proofs make use of a new equation that describes the behavior of maximal weight matching algorithms.

The rest of the paper is organized as follows. Section II introduces the terminology to model networks of queues and networks of switches. In section III, we define maximal weight matching scheduling policies, develop a mathematical model to describe their behavior and present stability results for networks of input-queued switches. We conclude in section IV.

2 Terminology and Model

2.1 Model of a network of queues

In this section, we follow an approach in [2] to describe our model of a queueing system. We assume a system of J physical queues $\tilde{q}^j$, $1 \leq j \leq J$ of infinite ca-

pacity. Each physical queue consists of one or more logical queues, where each logical queue corresponds to a certain class of customers within the physical queue. Whenever a packet moves from one physical queue to another, it changes class and therefore also changes logical queue. We denote a logical queue by q^k, $1 \leq k \leq K$, where $K \geq J$. A packet enters the network via an edge switch, travels through a number of switches and leaves the network via another edge switch. We define a function $L(k) = j$ that defines the physical queue $\tilde{q}^j$ at which packets belonging to the logical queue q^k are buffered. The inverse function $L^{-1}(j)$ returns the logical queues q^k that belong to the physical queue $\tilde{q}^j$.

Throughout this paper, the time t is described via a discrete, slotted time model. Packets are supposed to be of fixed size and a timeslot is the time needed by a packet to arrive completely at an input link.

We define a row vector $X_n = (x_n^1, ..., x_n^K)$, where the k-th vector x_n^k represents the number of packets buffered in the logical queue q^k at the beginning of the n-th timeslot. We define $E_n = (e_n^1, ..., e_n^K)$, where e_n^k equals the number of arrivals at the logical queue q^k in the n-th timeslot. Analogously, we define $D_n = (d_n^1, ..., d_n^K)$, where d_n^k expresses the number of departed packets from q^k in the n-th timeslot. We assume that packets arrive at a queue at the beginning of a timeslot and depart from a queue at the end of a timeslot. Thus, we can describe the dynamics of the system as follows:

$$X_{n+1} = X_n + E_n - D_n. \tag{1}$$

Packets that arrive at a logical queue q^k either arrive from outside the system or are forwarded from a queue within the system. Thus, we can write:

$$E_n = A_n + T_n, \tag{2}$$

where $A_n = (a_n^1, ..., a_n^K)$ denotes the arrivals from outside the system and $T_n = (t_n^1, ..., t_n^K)$ denotes the arrivals from inside the system.
We further define a routing matrix $R = [r_{i,j}]$, $1 \leq i, j \leq K$, where $r_{i,j}$ is the fraction of customers that depart from the logical queue q^i and are destined for the logical queue q^j. Assuming a deterministic routing policy, there holds, $r_{i,j} \in \{0, 1\}$, $\sum_{1 \leq i \leq K} r_{i,j} \leq 1$, $\sum_{1 \leq j \leq K} r_{i,j} \leq 1$. We set $r_{i,j} \neq 0$, if q^j follows q^i along the route. Noting that $T_n = D_n R$ and writing I for the identity diagonal matrix, we find from (1) and (**??**):

$$X_{n+1} = X_n + A_n - D_n(I - R). \tag{3}$$

We assume that the external arrival processes are stationary and satisfy the Strong Law of Large Numbers. Thus,

$$\lim_{n \to \infty} \frac{\sum_{i=1}^{n} A_i}{n} = \Lambda \qquad \text{w.p.1}, \tag{4}$$

where $E[A_n] = \Lambda = (\lambda^1, .., \lambda^K), \forall n \geq 1$[1].

We now calculate the average workload of the logical queues q^k which we

[1] Throughout the paper, we abbreviate "with probability 1" by "w.p.1".

denote by $W = (w^1, ..., w^K)$. The expected traffic arriving from outside the system is by definition equal to Λ. The traffic that arrives at the logical queues after having passed through m previous queues inside the network is by the definition of the routing matrix R equal to ΛR^m. Noting that $(I - R)^{-1} = I + R + R^2 + ...$, we find that the overall average workload at the logical queues q^k denoted as is given by $W = \Lambda(I - R)^{-1}$.

Finally, we give a stability criteria for a network of queues as proposed in [2].

Definition 1: A system of queues is rate stable if

$$\lim_{n \longrightarrow \infty} \frac{X_n}{n} = \lim_{n \longrightarrow \infty} \frac{1}{n} \sum_{i=0}^{n-1} (E_i - D_i) = 0 \qquad \text{w.p.1.}$$

A necessary condition for the rate stability of a system of queues is that the average number of packets that arrive at any physical queue $\tilde{q}^j$ during a timeslot is less than 1. In order to formalize this criteria, we introduce the following norm for a vector $Z \in \mathbb{R}^K$:

Definition 2: For a vector $Z \in \mathbb{R}^K$, $Z = (z^1, .., z^K)$, and the function $L^{-1}(k)$ as defined in this subsection, we set:

$$||Z||_{maxL} = \max_{j=1,..,J} \left\{ \sum_{k \in L^{-1}(j)} z^k \right\}. \tag{5}$$

If we apply this norm to the average workload vector W, then the expression $||W||_{maxL}$ denotes the maximum average workload over all physical queues $\tilde{q}^j$. The necessary condition for rate stability can now be formalized as follows:

$$||W||_{maxL} < 1. \tag{6}$$

In the sequel, we will say that a system of networks that satisfies condition (5) and is rate stable achieves 100% throughput.

2.2 Model of a network of switches

In this section, we apply the terminology of the previous section to a network of IQ/CIOQ switches. A network of IQ/CIOQ switches can be conceived as a queueing system as defined in the previous section where the virtual output queues are considered as the physical queues (The concept of virtual output queues is explained in the introduction of this article). In this model we neglect the output queues of the switches because instability can only occur at the VOQs (see [2]).

We say that packets that enter the network via the input of a given switch and leave the network via the output of a given switch belong to the same flow. Packets belonging to the same flow travel through the same sequence of physical queues and are mapped to the same logical queues at each physical queue, i.e. a flow can be mapped biunivocally to a series of logical queues.

We assume that each logical queue behaves as a FIFO queue and assume

a *per-flow* scheduling scheme which is more complex than a *per-virtual output queue* scheduling scheme. In sections III B - F, we prove the main results of this paper for *per-flow* scheduling schemes. In [2], it has been shown how *per flow* scheduling schemes can be used to design *per-virtual output queue* scheduling schemes.

The network consists of B switches and each switch has N_b, $1 \leq b \leq B$, inputs and outputs. If the total number of flows in the system is T, we do not have more than N_b^2 physical queues and TN_b^2 logical queues at switch b. We can model the whole network of switches as a system of $\sum_{1 \leq b \leq B} TN_b^2$ logical queues. For the sake of simplicity, we suppose that $N_b = N$, $\forall b$, $1 \leq b \leq B$ and set $K = TN^2B$. Finally, we define $Q_I(b,i)$ as the set of indexes corresponding to the logical queues at the i-th input of the b-switch. Analogously, $Q_O(b,i)$ denotes the set of indexes corresponding to the logical queues directed to the i-th output of the b-switch. We further note that logical queues are defined per switch, per virtual-output queue and per flow. Thus, the index k of any logical queue in the network can be uniquely expressed as $k = TN^2b + TNi + Tj + l$, $0 \leq b < B$, $0 \leq i, j < N, 0 \leq l < T$. We use these definitions to adapt the norm $||Z||_{maxL}$ to a network of switches that handle multiple flows at their inputs.

Definition 3: Given a vector $Z \in \mathbb{R}^K, Z = \{z^k, k = TN^2b + TNi + Tj + l$, $0 \leq b < B$, $0 \leq i, j < N, 0 \leq l < T$, the norm $||Z||_{IO}$ is defined as follows:

$$||Z||_{IO} = \max_{\substack{b=1,..,B \\ i=1,..,N}} \left\{ \sum_{m \in Q_I(b,i)} |z^m|, \sum_{m \in Q_O(b,i)} |z^m| \right\}.$$

Because we assume a deterministic routing policy, the necessary condition for rate stability given in (5) can now be written for a network of switches as follows:

Definition 4: For a network of IQ/CIOQ switches, a traffic and routing pattern W is admissible if and only if:

$$||W||_{IO} = ||\Lambda(I - R)^{-1}|| < 1. \tag{7}$$

Without further mentioning, in the rest of this paper, we will only consider traffic and routing patterns that satisfy the condition (6).

2.3 Weight function

All scheduling policies introduced in this paper are matching policies with specific weights. The weights are defined using a family of real positive vector functions $f(x) = \{f_k(x) : \mathbb{N} \to \mathbb{R}, 1 \leq k \leq K, \}$ that satisfy the following property:

$$\lim_{n \to \infty} \frac{f_k(n)}{n} = \frac{1}{w^k} \qquad \text{w.p.1.} \tag{8}$$

We define $\overline{d}^k(n) = \sum_{m \leq n} d_m^k$ as the cumulative number of services at queue q^k up to time n. For a given positive constant C, we define the weight of the queue q^k

at time n as

$$\phi_n^k(f) := \phi_n^k = n - f_k(\overline{d}^k(n)) + C. \tag{9}$$

We set $\Phi_n(f) := \Phi_n = (\phi_n^1, .., \phi_n^K)$. We see that for a fixed function $f_k(\cdot)$ that satisfies the relation (7), there exists a constant $C > 0$ such that $\forall n$, $n \geq 0$, there holds $f_k(\overline{d}_n^k) \leq \frac{\overline{d}_n^k}{w^k} - C$. Further, because the accumulative departure rate at each logical queue q^k cannot be more than the accumulative arrival rate, there holds $\lim_{n\to\infty} \frac{\overline{d}^k(n)}{w^k} \leq n$. Combining these two estimates, we see that for any function $f(\cdot)$ that satisfies the condition (7), one can always find a $C > 0$ such that the weights ϕ_n^k are positive $\forall n$, $n \geq 0$, $\forall k$, $1 \leq k \leq K$. This fact is important for the theorem 2 and 3 below, which all require the weights ϕ_n^k to be strictly positive.

In [2], an example for $f_k(n)$ is given. The cumulative function of external arrivals for the logical queue q^k is given by $\overline{a}^k(n) = \sum_{m \leq n} a_m^k$. The inverse function $[\overline{a}^k]^{-1}(p)$ maps the packet number p to the arrival slot. Setting $C = 0$ and $f_k(p) = [\overline{a}^k]^{-1}(p)$, the weight $\phi_n^k = n - [\overline{a}^k]^{-1}(p)$ denotes the age of the p-th packet at time n. At its departure time n, the age of the p-th packet is $n - [\overline{a}^k]^{-1}(\overline{d}_n^k)$.

2.4 Maximal weight matching algorithms

In this section, we propose a maximal weight matching scheduling policy for a network of switches with a speedup of $S > H$. We suppose that packets arrive at the beginning of an external timeslot t and are transferred instantly at the end of an internal timeslot. As the switching core is modeled as a crossbar, in every internal timeslot at most one packet can be sent from the same input or to the same output, i.e.

$$||D_n||_{IO} \leq S, \qquad \forall n \geq 1. \tag{10}$$

Further, for a given input i and a given output j at a given switch b, we define the set of all logical queues that either belong to the input i or that are directed to the output j. We set $\forall b, i, j$, $1 \leq b \leq B$, $1 \leq i, j \leq N$,

$$\mathcal{S}_{b,i,j} := \left\{ m : \, 1 \leq m \leq K, \, m \in Q_I(b,i) \bigcup Q_O(b,j) \right\}.$$

For each set $\mathcal{S}_{b,i,j}$, we sum the average arrival rates for all logical queues that belong to $\mathcal{S}_{b,i,j}$, and define the maximum of these summations as H :

$$H = \max_{\substack{1 \leq b \leq B \\ 1 \leq i,j \leq N}} \sum_{m \in \mathcal{S}_{b,i,j}} w^m. \tag{11}$$

From (6), we see $H < 2$. For a set of positive weights P^k, $1 \leq k \leq K$, where P^k is the weight assigned to the logical queue q^k, we now formally define a maximal weight matching algorithm as follows:

1. Initially, all logical queues q^k, $1 \leq k \leq K$, are considered potential choices for a cell transfer.
2. The logical queue with the largest weight, say q^{k_0} is chosen for a cell transfer and ties are broken randomly. We assume without loss of generality that $k_0 \in Q_I(b_1, i_1)$ and $k_0 \in Q_O(b_1, j_1)$.
3. All logical queues q^k with $k \in \mathcal{S}_{b_1,i_1j_1}$ are removed.
4. If all q^k are removed, the algorithm terminates. Else go to step 2.

Now, we establish a lower bound for the weight of a matching calculated by a maximal weight matching algorithm. This bound will be used to prove the stability of a specific maximal weight matching scheduling policy in theorem 2. We define $P_n = (P_n^1, ..., P_n^K)$ as the weight of logical queue q^k at the beginning of the n-th external timeslot and $P_{n,s}^k$, $1 \leq s \leq S$, as the weight of q^k at the beginning of the s-th internal timeslot of the n-th external timeslot. Thus, $P_n = P_{n,1}$, $\forall\, n \geq 0$. We also define the departure vector of the s-th internal timeslot of the n-th external timeslot as $D_{n,s} = (d_{n,s}^1, ..., d_{n,s}^K)$ such that $D_n = \sum_{s=1}^{S} D_{n,s}$, and $\forall n \geq 1$, $\forall 1 \leq s \leq S$:

$$||D_{n,s}||_{IO} \leq 1. \tag{12}$$

Now, we show that the weight of a matching calculated in an external timeslot is $1/H$-times larger than the sum of the products of the average arrival rate w^k of a logical queue q^k multiplied with the actual weight $P_{n,s}^k$ summed over all logical queues and all internal timeslot of the considered external timeslot.

Theorem 1: *For a network of IQ/CIOQ switches that applies a maximal weight matching algorithm for positive weights $P^k > 0, 1 \leq k \leq K$, and a speedup-up of S, there holds for any timeslot n*

$$\frac{1}{H} \sum_{s=1}^{S} \sum_{k=1}^{K} P_{n,s}^k w^k \leq \sum_{k=1}^{K} \sum_{s=1}^{S} P_{n,s}^k d_{n,s}^k.$$

Proof: We first analyze a maximal weight matching algorithm in the first internal timeslot. In its first iteration, the algorithm selects the queue with the largest weight, say $P_n^{k_1}$ with $k_1 \in \mathcal{S}_{b_1,i_1,j_1}$, for transfer. As $P^k > 0$, we see from (10):

$$P_n^{k_1} H \geq P_n^{k_1} \sum_{m \in \mathcal{S}_{b_1,i_1,j_1}} w^m \geq \sum_{m \in \mathcal{S}_{b_1,i_1,j_1}} P_n^m w^m.$$

All logical queues q^k with $k \in \mathcal{S}_{b_1,i_1,j_1}$ are removed. In the second iteration, the remaining queue with the largest weight, say $P_n^{k_2}, k_2 \in \mathcal{S}_{b_2,i_2,j_2}$ is chosen. Thus,

$$P_n^{k_2} H \geq P_n^{k_2} \sum_{\substack{m \in \mathcal{S}_{b_2,i_2,j_2} \\ m \notin \mathcal{S}_{b_1,i_1,j_1}}} w^m \geq \sum_{\substack{m \in \mathcal{S}_{b_2,i_2,j_2} \\ m \notin \mathcal{S}_{b_1,i_1,j_1}}} P_n^m w^m. \tag{13}$$

The matching algorithm stops after h, $h \leq BN$ iterations when none or only empty queues remain. For each of these h iterations, an inequality analogous to

(12) holds. If any empty queues remain, we hypothetically continue to run the algorithm $(BN - h)$ times and produce valid inequalities as in (12) where both sides are equal to zero. Summing over all BN inequalities, we obtain

$$\sum_{m=1}^{BN} P_n^{k_m} = \sum_{k=1}^{K} P_n^k d_n^k \geq \frac{1}{H} \sum_{k=1}^{K} P_n^k w^k. \tag{14}$$

Applying this analysis to all S internal timeslots, we obtain from (13)

$$\sum_{s=1}^{S} \sum_{k=1}^{K} P_{n,s}^k d_{n,s}^k \geq \frac{1}{H} \sum_{s=1}^{S} \sum_{k=1}^{K} P_{n,s}^k w^k. \qquad \square$$

The next corollary can be derived from (6), (11) and theorem 1:

Corollary 1 *If under the assumptions of theorem 1 there holds for a positive constant* C_1, $|P_{n,s}^k - P_{n,s+1}^k| \leq C_1$ $1 \leq k \leq K$, $n \geq 1$, $1 \leq s \leq S-1$, *then:*

$$\frac{S}{H} \sum_{k=1}^{K} P_n^k(t) w^k \leq \sum_{k=1}^{K} P_n^k d_n^k + \frac{S(S-1)}{2} K C_1 \left(1 + \frac{1}{H}\right).$$

The assumption of corollary 1 that weights only change by a bounded amount between two iterations of the algorithm is valid for most weight functions proposed in the literature, and in particular for the weight function defined in (8).

The main results of this paper states the existence of maximal weight matching scheduling policies that guarantee 100% throughput of the network:

Theorem 2: *A network of IQ/CIOQ switches that implements a single maximal weight matching scheduling policy with a speedup of* $S > H$ *with the weights* $\Phi_n > 0$ *at time* n *defined as in (8), achieves 100% throughput.*

Moreover, a network of switches in which individual switches deploy different scheduling policies $\mathcal{P}^m$ defined with regard to different weight functions $\Phi_n(f^m)$ in (8), also achieves 100% throughput.

Theorem 3 *Under the conditions of theorem 2, a network of IQ/CIOQ switches where each switch deploys any policy* $\mathcal{P}^m$ *achieves 100% throughput.*

Remark: The condition (9) and the steps 2 and 3 of a maximal weight matching algorithm show that scheduling decisions taken at different switches are independent of each other. Thus, in a distributed implementation, each switch executes the algorithm on the set of logical queues that belong to its physical queues.

2.5 Proofs of theorems 2 and 3

For the proof of theorem 2, we introduce the fluid methodology as applied in [6]. We denote the set of all switch configurations found at time t by maximal weight matching algorithms as defined in theorem 2 as $\prod_1(t)$. Using (2), we obtain the fluid equations of the system as follows:

$$X(t) = X(0) + \Lambda t - D(t)(I - R), \tag{15}$$

$$D(t) = \sum_{\pi_1 \in \prod_1(t)} \pi_1 T_{\pi_1}(t), \tag{16}$$

$$\sum_{\pi_1 \in \prod_1(t)} T_{\pi_1}(t) = t. \tag{17}$$

where $T_{\pi_1}(t)$ is a non-decreasing function denoting the cumulative amount of time that the matching π_1 has been used up to time t. Also, noting that by (7) $\lim_{t\to\infty} f_k(t) \to t/w^k$, and that $\overline{d}^k(t) \to \infty$ for $t \to \infty$, we obtain

$$\phi^k(t) \to t - \frac{\overline{d}^k(t)}{w^k} + C. \tag{18}$$

We define $\mathit{\Gamma} = [\gamma^{(i,j)}]$ as the diagonal matrix with $\gamma^{(k,k)} = w^k$, and let $\mathit{\Gamma}^{-1}$ be the inverse of $\mathit{\Gamma}$. We see from (17):

$$\dot{\mathit{\Phi}}(t) = \mathbb{I} - \dot{D}(t)\mathit{\Gamma}^{-1}. \tag{19}$$

Writing the scalar product for two vectors v_1 and v_2 as $\langle v_1, v_2\rangle = v_1 v_2^T$, we define the Lyapunov function $V(\mathit{\Phi}(t)) = \frac{1}{2}\langle \mathit{\Phi}(t)\mathit{\Gamma}, \mathit{\Phi}(t)\rangle$. We want to show that $\forall t \geq 0$,

$$\mathit{\Phi}(t) \leq B, \tag{20}$$

for a certain constant $B > 0$. As $V(\mathit{\Phi}(0)) = \langle \mathit{\Phi}(0)\mathit{\Gamma}, \mathit{\Phi}(0)\rangle \geq 0$, we see that if

$$\frac{d}{dt} V(\mathit{\Phi}(t)) \leq 0 \tag{21}$$

$\forall t$ such that $\langle \mathbb{I}, \mathit{\Phi}(t)\rangle \geq K_0$ for a fixed $K_0 > 0$, then $\forall t \geq 0$, there holds $V(\mathit{\Phi}(t)) \leq V(L_0) := \max_{\substack{L\in\mathbb{R}^K,\\ \langle \mathbb{I}, L\rangle \leq K_0}} V(L)$, which in turn implies (19) for a certain $B > 0$. For the proof of (20), we note that for any t, there holds by the pigeonhole principle:

$$\max_{1\leq k\leq K} \phi^k(t) \geq \frac{\langle \mathbb{I}, \mathit{\Phi}(t)\rangle}{K}. \tag{22}$$

Finally, we express corollary 1 in the following way:

$$\frac{S}{H}\langle W, \mathit{\Phi}(t)\rangle \leq \langle \pi_1, \mathit{\Phi}(t)\rangle + K_1, \text{ where } K_1 = \frac{S(S-1)}{2} K C_1 (1 + \frac{1}{H}), \tag{23}$$

$\forall \pi_1 \in \mathit{\Pi}_1(t)$. Now (20) follows from (15), (16), (18), (21), and (22):

$$\begin{aligned}
&\frac{d}{dt} V(\mathit{\Phi}(t)) = \frac{1}{2}\langle \dot{\mathit{\Phi}}(t)\mathit{\Gamma}, \mathit{\Phi}(t)\rangle + \frac{1}{2}\langle \mathit{\Phi}(t)\mathit{\Gamma}, \dot{\mathit{\Phi}}(t)\rangle \\
&= \langle \dot{\mathit{\Phi}}(t)\mathit{\Gamma}, \mathit{\Phi}(t)\rangle = \langle [\mathbb{I} - \dot{D}(t)\mathit{\Gamma}^{-1}]\mathit{\Gamma}, \mathit{\Phi}(t)\rangle \\
&= \langle W, \mathit{\Phi}(t)\rangle - \sum_{\pi_1 \in \prod_1(t)} \langle \pi_1 \dot{T}_{\pi_1}(t), \mathit{\Phi}(t)\rangle \\
&= \langle W, \mathit{\Phi}(t)\rangle - \sum_{\pi_1 \in \prod_1(t)} \dot{T}_{\pi_1}(t)\langle \pi_1, \mathit{\Phi}(t)\rangle
\end{aligned}$$

$$\leq \langle W, \Phi(t)\rangle \left(1 - \frac{S}{H}\right) + K_1 \leq \left(1 - \frac{S}{H}\right) \min_{\substack{1\leq k\leq K \\ w^k>0}} w^k \frac{\langle \mathbb{I}, \Phi(t)\rangle}{K} + K_1$$

$$\leq \frac{H-S}{2H} \min_{\substack{1\leq k\leq K \\ w^k>0}} w^k \frac{\langle \mathbb{I}, \Phi(t)\rangle}{K} < 0,$$

where the second to last inequality holds for $\langle \mathbb{I}, \Phi(t)\rangle > K_0 > 0$ if K_0 is chosen sufficiently large. We see from (17) and (19): $0 < t - \frac{\overline{d}^k(t)}{w^k} + C \leq B$, which implies

$$\lim_{t\to\infty} \frac{D(t)}{t} = W, \qquad \text{w.p.1,} \tag{24}$$

which corresponds to the rate stability condition of $X(t)$. For the proof of theorem 3, we divide the switches in the network into M groups G_m, $1 \leq m \leq M$, where each group G_m contains the switches that deploy the switching policy $\mathcal{P}^M$ Accordingly, we can divide the departure vector $D(t)$ and the arrival rate vector W in M subvectors, $D_m(t)$ and W_m. In order to prove the stability condition (23), it is sufficient to show that for each m $\lim_{t\to\infty} D_m(t)/t = W_m$, w.p.1. For each m, this relation can be proved by applying the proof of theorem 2 to the group G_m instead of applying it to the whole network of switches.

3 Conclusions

This paper investigates distributed scheduling policies of low complexity for networks of input-queued switches. It defines a class of scheduling policies based on maximal weight matching algorithms that guarantee the stability of networks where all switches deploy the same scheduling policy. It is also shown that a network where each switch deploys any policy out of the defined class of switching policies is stable.

References

1. Ajmone, M.M.,Leonardi, E., Mellia, M., Neri, F., *On the throughput achievable by isolated and interconnected input-queued switches under multicalss traffic,* Proc. of Infocom 2002, New York City, June 2002.
2. Ajmone, M.M.,Giaccone, P., Leonardi, E., Mellia, M., Neri, F., *Local scheduling policies in networks of packet switches with input queues,* Proc. of Infocom 2003, San Francisco, April 2003.
3. Andrews, M., Zhang, L., *Achieving stability in networks of input queued,* Proc. of Infocom 2001, Anchorage, Alaska, April 2001.
4. Bauer, C., *Packet scheduling in input-queued switches with a speedup of less than two,* Proc. of IEEE International Conference on Networks, Sydney, Sept. 2003.
5. Benson, K., *Throughput of crossbar switches using maximal matching algorithms,* Proc. of IEEE ICC 2002, New York City.
6. Dai, J.G., Prabhakar, B., *The throughput of data switches with and without speedup,* Proc. of IEEE Infocom 2000, Tel Aviv.

7. Leonardi, E., Mellia, M., Neri, F., Marsan, M.A., *Stability of maximal size matching scheduling in input queued cell switches,* Prof. of IEEE ICC 2000, New Orleans.
8. McKeown, N., Mekkittikul, A., Anantharam, V., Walrand, J., *Achieving 100% throughput in an input queued switch,* IEEE Transactions on Communications, vol. 47, no. 8, Aug. 1999, 1260 - 1272.
9. Shah, D., Kopikare, M., *Delay bounds for approximate maximum weight matching algorithms for input queued switches,* Proc. of IEEE Infocom 2002, New York City, June 2002.