# QPS-r: A Cost-Effective Iterative Switching Algorithm for Input-Queued Switches

Long Gong
Georgia Tech
gonglong@gatech.edu

Jun (Jim) Xu
Georgia Tech
jx@cc.gatech.edu

Liang Liu
Georgia Tech
liuliang142857@gatech.edu

Siva Theja Maguluri
Georgia Tech
siva.theja@gatech.edu

## ABSTRACT

In an input-queued switch, a crossbar schedule, or a matching between the input ports and the output ports needs to be computed for each switching cycle, or time slot. It is a challenging research problem to design switching algorithms that produce high-quality matchings yet have a very low computational complexity when the switch has a large number of ports. Indeed, there appears to be a fundamental tradeoff between the computational complexity of the switching algorithm and the quality of the computed matchings.

Parallel maximal matching algorithms (adapted for switching) appear to be a sweet tradeoff point in this regard. On one hand, they provide the following performance guarantees: Using maximal matchings as crossbar schedules results in at least 50% switch throughput and *order-optimal* (*i.e.,* independent of the switch size $N$) average delay bounds for various traffic arrival processes. On the other hand, their computational complexities can be as low as $O(\log^2 N)$ per port/processor, which is much lower than those of the algorithms for finding matchings of higher qualities such as maximum weighted matching.

In this work, we propose QPS-r, a parallel iterative switching algorithm that has the lowest possible computational complexity: $O(1)$ per port. Yet, the matchings that QPS-r computes have the same quality as maximal matchings in the following sense: Using such matchings as crossbar schedules results in exactly the same aforementioned provable throughput and delay guarantees as using maximal matchings, as we show using Lyapunov stability analysis. Although QPS-r builds upon an existing add-on technique called Queue-Proportional Sampling (QPS), we are the first to discover and prove this nice property of such matchings. We also demonstrate that QPS-3 (running 3 iterations) has comparable empirical throughput and delay performances as iSLIP (running $\log_2 N$ iterations), a refined and optimized representative maximal matching algorithm adapted for switching.

## CCS CONCEPTS

• **Mathematics of computing** → **Markov processes**.

**ACM Reference Format:**
Long Gong, Jun (Jim) Xu, Liang Liu, and Siva Theja Maguluri. 2020. QPS-r: A Cost-Effective Iterative Switching Algorithm for Input-Queued Switches. In *13th EAI International Conference on Performance Evaluation Methodologies and Tools (VALUETOOLS '20), May 18–20, 2020, Tsukuba, Japan.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3388831.3388836


*VALUETOOLS '20, May 18–20, 2020, Tsukuba, Japan*

ACM ISBN 978-1-4503-7646-4/20/05…$15.00
https://doi.org/10.1145/3388831.3388836

## 1 INTRODUCTION

Many present day high-performance switching systems in Internet routers and data-center switches employ an input-queued crossbar to interconnect their input ports and output ports. In an $N \times N$ input-queued crossbar switch, each input port can be connected to only one output port and vice versa in each switching cycle or time slot. Hence, in every time slot, the switch needs to compute a one-to-one *matching* between input and output ports (*i.e.*, the crossbar schedule). A major research challenge in designing high-link-rate switches with a large number of ports (called *high-radix* [2, 3]) is to develop algorithms that can compute "high quality" matchings – *i.e.,* those that result in high switch throughput and low queueing delays for packets – in a few nanoseconds. Clearly, a suitable switching algorithm has to have very low computational complexity, yet output "fairly good" matching decisions most of time.

### 1.1 The Family of Maximal Matchings

A family of parallel iterative algorithms for computing *maximal matching* (one to which no edge can be added for it to remain a matching, a definition that will be made precise in §2) are arguably the best candidates for switching in high-link-rate high-radix switches, because they have reasonably low computational complexities, yet can provide fairly good throughput and delay performance guarantees. More specifically, using maximal matchings as crossbar schedules results in at least 50% switch throughput in theory (and usually much higher throughput in practice) [4]. In addition, it results in low packet delays that also have excellent scaling behaviors such as *order-optimal* (*i.e.,* independent of switch size $N$) under various traffic arriving processes when the offered load is less than 50% (*i.e.*, within the provable stability region) [17]. In comparison, matchings of higher qualities such as maximum matching (with the largest possible number of edges) and maximum weighted matching (with the highest total edge weight) are much more expensive to compute. Hence, it is fair to say that, maximal matching algorithms overall deliver the biggest "bang" (performance) for the "buck" (computational complexity).

Unfortunately, parallel maximal matching algorithms are still not "dirt cheap" computationally. More specifically, all existing parallel algorithms that compute maximal matchings on *general* $N \times N$ bipartite graphs require a minimum of $O(\log N)$ iterations (rounds of message exchanges). This minimum is attained by the classical algorithm of Israel and Itai [12]; the PIM algorithm [1] is a slight adaptation of this classical algorithm to the switching context, and iSLIP [15] further improves upon PIM by reducing its

per-iteration per-port computational complexity to $O(\log N)$ via de-randomizing a computationally expensive ($O(N)$ complexity to be exact) operation in PIM.

### 1.2 QPS-r: Bigger Bang for the Buck

In this work, we propose QPS-r, a parallel iterative switching algorithm that has the lowest possible computational complexity: $O(1)$ per port. More specifically, QPS-r requires only $r$ (a small constant independent of $N$) iterations to compute a matching, and the computational complexity of each iteration is only $O(1)$; here QPS stands for Queue-Proportional Sampling, an add-on technique proposed in [8] that we will describe shortly. Yet, even the matchings that QPS-1 (running only a single iteration) computes have the same (reasonably high) quality as maximal matchings in the following sense: Using such matchings as crossbar schedules results in exactly the same aforementioned provable throughput and delay guarantees as using maximal matchings, as we will show using Lyapunov stability analysis. QPS-r performs as well as maximal matching algorithms not just in theory: We will show in §5 that QPS-3 (running 3 iterations) has comparable empirical throughput and delay performances as iSLIP (running $\log_2 N$ iterations), a refined and optimized representative maximal matching algorithm adapted for switching, under various workloads. Note that matchings that QPS-r computes are generally not maximal. QPS-r can make do with less (iterations) because the queue-proportional sampling operation implicitly makes use of the queue (VOQ) length information, which maximal matching algorithms do not. One major contribution of this work is to discover the family of (QPS-r)-generated matchings that is even more cost-effective.

Although QPS-r builds on the QPS data structure and algorithm proposed in [8], our work on QPS-r is very different in three important aspects. First, in [8], QPS was used only as an add-on to other switching algorithms such as iSLIP [15] and SERENA [6] by generating a starter matching for other switching algorithms to further refine, whereas in this work, QPS-r is used only as a stand-alone algorithm. Second, we are the first to discover and prove that (QPS-r)-generated matchings and maximal matchings provide exactly the same aforementioned performance guarantees, whereas in [8], no such mathematical similarity or connection was mentioned. Third, the establishment of this mathematical similarity is an important theoretical contribution in itself, because maximal matchings have long been established as a cost-effective family both in switching [1, 15] and in wireless networking [17, 18], and with this connection we have considerably enlarged this family.

Although we show that QPS-r has exactly the same throughput and delay bounds as that of maximal matchings established in [4, 17, 18], our proofs are different for the following reason. A *departure inequality* (see Property 1), satisfied by all maximal matching algorithms was used in the throughput analysis of [4] and the delay analysis of [17, 18]. This inequality, however, is not satisfied by QPS-r in general. Instead, QPS-r satisfies this departure inequality in expectation, which is a much weaker guarantee. A methodological contribution of this work is to prove two theorems stating that this much weaker guarantee is sufficient for obtaining the same throughput and delay bounds respectively.

The rest of this paper is organized as follows. In §2, we provide some background on the switching problem in input-queued crossbar switches. In §3, we first review QPS, and then describe QPS-r. Then in §4, we derive the throughput and the queue length (and delay) bounds of QPS-r, followed by the performance evaluation in §5. In §6, we survey related work before concluding this paper in §7.

## 2 BACKGROUND

In this section, we provide a brief introduction to the switching (crossbar scheduling) problem. Throughout this paper, we adopt the following standard assumption that all the incoming variable-size packets are first segmented into fixed-size packets (also referred to as cells), and then reassembled at their respective output ports before leaving the switch. In the sequel, we consider the switching of only fixed-size packets, and each fixed-size cell takes one time slot to switch. We also assume that all input links/ports and output links/ports operate at the same normalized line rate of 1, and so do all wires and crosspoints inside the crossbar.

In an $N \times N$ input-queued crossbar switch, each input port has $N$ Virtual Output Queues (VOQs) [20]. The $j^{th}$ VOQ at input port $i$ serves as a buffer for packets going from input port $i$ to output port $j$. The use of VOQs solves the Head-of-Line (HOL) blocking issue [13], which severely limits the throughput of the switch system.

An $N \times N$ input-queued crossbar can be modeled as a bipartite graph, of which the two disjoint vertex sets are the $N$ input ports and the $N$ output ports respectively. In this bipartite graph, there is an edge between input port $i$ and output port $j$, if and only if the $j^{th}$ VOQ at input port $i$, the corresponding VOQ, is nonempty. A set of such edges constitutes a *valid crossbar schedule*, or a *matching*, if any two of them do not share a common vertex. A matching $M$ is called a *maximal matching*, if it is no longer a matching, when any edge not in $M$ is added to it.

A matching $M$ can be represented as an $N \times N$ *sub-permutation matrix* (a 0-1 matrix that contains at most one entry of "1" in each row and in each column) $S = (s_{ij})$ as follows: $s_{ij} = 1$ if and only if the edge between input port $i$ and output port $j$ is contained in $M$ (*i.e.,* input port $i$ is matched to output port $j$ in $M$). To avoid any confusion, only $S$ (not $M$) is used to denote a matching in the sequel, and it can be both a set (of edges) and a matrix.

## 3 THE QPS-$r$ ALGORITHM

The QPS-r algorithm simply runs $r$ iterations of QPS (Queue-Proportional Sampling) [8] to arrive at a matching, so its computational complexity per port is exactly $r$ times those of QPS. Since $r$ is a small constant, it is $O(1)$, same as that of QPS. In the following two subsections, we describe QPS and QPS-r respectively in more details.

### 3.1 Queue-Proportional Sampling (QPS)

QPS was used in [8] as an "add-on" to augment other switching algorithms as follows. It generates a starter matching, which is then populated (*i.e.*, adding more edges to it) and refined, by other switching algorithms such as iSLIP [15] and SERENA [6], into a final matching. To generate such a starter matching, QPS needs to run only one iteration, which consists of two phases, namely, a

proposing phase and an accepting phase. We briefly describe them in this section for this paper to be self-contained.

*3.1.1 The Proposing Phase.* In this phase, each input port proposes to *exactly one* output port – decided by the QPS strategy – unless it has no packet to transmit. Here we will only describe the operations at input port 1; that at any other input port is identical. Like in [8], we denote by $m_1, m_2, \cdots, m_N$ the respective queue lengths of the $N$ VOQs at input port 1, and by $m$ their total (*i.e.*, $m \triangleq \sum_{k=1}^{N} m_k$). Input port 1 simply samples an output port $j$ with probability $\frac{m_j}{m}$, *i.e.*, proportional to $m_j$, the length of the corresponding VOQ (hence the name QPS); it then proposes to output port $j$, with the value $m_j$ that will be used in the next phase. The computational complexity of this QPS operation, carried out using a simple data structure proposed in [8], is $O(1)$ per (input) port.

*3.1.2 The Accepting Phase.* We describe only the action of output port 1 in the accepting phase; that of any other output port is identical. The action of output port 1 depends on the number of proposals it receives. If it receives exactly one proposal from an input port, it will accept the proposal and match with the input port. However, if it receives proposals from multiple input ports, it will accept the proposal accompanied with the largest VOQ length (called the "longest VOQ first" accepting strategy), with ties broken uniformly at random. The computational complexity of this accepting strategy is $O(1)$ on average and can be made $O(1)$ even in the worst case [8].

## 3.2 The QPS-r Scheme

The QPS-r scheme simply runs $r$ QPS iterations. In each iteration, each input port that is not matched yet, first proposes to an output port according to the QPS proposing strategy; each output port that is not matched yet, accepts a proposal (if it has received any) according the "longest VOQ first" accepting strategy. Hence, if an input port has to propose multiple times (once in each iteration), due to all its proposals (except perhaps the last) being rejected, the identities of the output ports it "samples" (*i.e.,* proposes to) during these iterations are samples with replacement, which more precisely are *i.i.d.* random variables with a queue-proportional distribution.

At the first glance, sampling with replacement may appear to be an obviously suboptimal strategy for the following reason. There is a nonzero probability for an input port to propose to the same output port multiple times, but since the first (rejected) proposal implies this output port has already accepted "someone else" (a proposal from another input port), all subsequent proposals to this output port will surely go to waste. For this reason, sampling without replacement (*i.e.*, avoiding all output ports proposed to before) may sound like an obviously better strategy. However, it is really not, since compared to sampling with replacement, it has a much higher computational complexity of $O(\log N)$, but improves the throughput and delay performances only slightly according to our simulation studies.

# 4 THROUGHPUT AND DELAY ANALYSIS

In this section, we show that QPS-1 (*i.e.*, running a single QPS iteration) delivers exactly the same provable throughput and delay guarantees as maximal matching algorithms. When $r > 1$, QPS-r clearly should have better throughput and delay performances than QPS-1, as more input and output ports can be matched up during subsequent iterations, although we are not able to derive stronger bounds.

## 4.1 Preliminaries

In this section, we introduce the notation and assumptions that will later be used in our derivations. We define three $N \times N$ matrices $Q(t), A(t)$, and $D(t)$. Let $Q(t) \triangleq \big(q_{ij}(t)\big)$ be the queue length matrix where each $q_{ij}(t)$ is the length of the $j^{th}$ VOQ at input port $i$ during time slot $t$. With a slight abuse of notation, we refer to this VOQ as $q_{ij}$ (without the $t$ term).

We define $Q_{i*}(t)$ and $Q_{*j}(t)$ as the sum of the $i^{th}$ row and the sum of the $j^{th}$ column respectively of $Q(t)$, *i.e.*, $Q_{i*}(t) \triangleq \sum_j q_{ij}(t)$ and $Q_{*j}(t) \triangleq \sum_i q_{ij}(t)$. With a similar abuse of notation, we define $Q_{i*}$ as the VOQ set $\{q_{i1}, q_{i2}, \cdots, q_{iN}\}$ (*i.e.*, those on the $i^{th}$ row), and $Q_{*j}$ as $\{q_{1j}, q_{2j}, \cdots, q_{Nj}\}$ (*i.e.*, those on the $j^{th}$ column).

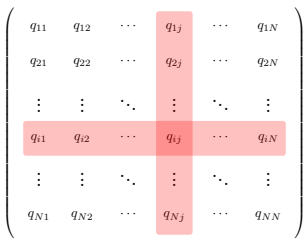

**Figure 1: $Q^{\dagger}_{ij}$: neighborhood of $q_{ij}$.**

Now we introduce a concept that lies at the heart of our derivations: neighborhood. For each VOQ $q_{ij}$, we define its neighborhood as $Q_{i*} \bigcup Q_{*j}$, the set of VOQs on the $i^{th}$ row or the $j^{th}$ column. We denote this neighborhood as $Q^{\dagger}_{ij}$, since it has the shape of a cross. Figure 1 illustrates $Q^{\dagger}_{ij}$, where the row and column in the shadow are the VOQ sets $Q_{i*}$ and $Q_{*j}$ respectively. $Q^{\dagger}_{ij}$ can be viewed as the *interference set* of VOQs for VOQ $q_{ij}$ [17, 18], as no other VOQ in $Q^{\dagger}_{ij}$ can be active (*i.e.,* transmit packets) simultaneously with $q_{ij}$. We define $Q^{\dagger}_{ij}(t)$ as the total length of all VOQs in (the set) $Q^{\dagger}_{ij}$ at time slot $t$, that is

$$Q^{\dagger}_{ij}(t) \triangleq Q_{i*}(t) - q_{ij}(t) + Q_{*j}(t). \tag{1}$$

Here we need to subtract the term $q_{ij}(t)$ so that it is not double-counted (in both $Q_{i*}(t)$ and $Q_{*j}(t)$).

Let $A(t) = \big(a_{ij}(t)\big)$ be the traffic arrival matrix where $a_{ij}(t)$ is the number of packets arriving at the input port $i$ destined for output port $j$ during time slot $t$. For ease of exposition, we assume that, for each $1 \le i, j \le N$, $\{a_{ij}(t)\}_{t=0}^{\infty}$ is a sequence of *i.i.d.* random variables, the second moment of their common distribution ($= \mathbf{E}\big[a^2_{ij}(0)\big]$) is finite, and this sequence is independent of other sequences (for a different $i$ and/or $j$). Our analysis, however, holds for more general arrival processes (*e.g.*, Markovian arrivals) that were considered in [17, 18], as we will elaborate shortly. Let $D(t) = \big(d_{ij}(t)\big)$ be the departure matrix for time slot $t$ output by the switching algorithm. Similar to $S$, $D(t)$ is a 0-1 matrix in which $d_{ij}(t) = 1$ if and only if a packet departs from $q_{ij}$ during time slot $t$. For any $i, j$, the queue length process $q_{ij}(t)$ evolves as follows:

$$q_{ij}(t+1) = q_{ij}(t) - d_{ij}(t) + a_{ij}(t). \tag{2}$$

Let $\Lambda = \big(\lambda_{ij}\big)$ be the (normalized) traffic rate matrix (associated with $A(t)$) where $\lambda_{ij}$ is normalized (to the percentage of the line

rate of an input/output link) mean arrival rate of packets to VOQ $q_{ij}$. With $a_{ij}(t)$ being an *i.i.d.* process, we have $\lambda_{ij} = \mathbf{E}\big[a_{ij}(0)\big]$. We define $\rho_\Lambda$ as the maximum load factor imposed on any input or output port by $\Lambda$,

$$\rho_\Lambda \triangleq \max\Big\{ \max_{1\le i\le N}\{\sum_j \lambda_{ij}\}, \max_{1\le j\le N}\{\sum_i \lambda_{ij}\}\Big\} \tag{3}$$

A switching algorithm is said to achieve 100% throughput or be throughput-optimal if the (packet) queues are stable whenever $\rho_\Lambda < 1$.

As mentioned before, we will prove in this section that, same as the maximal matching algorithms, QPS-1 is stable *under any traffic arrival process* $A(t)$ whose rate matrix $\Lambda$ satisfies $\rho_\Lambda < 1/2$ (*i.e.*, can provably attain at least 50% throughput, or half of the maximum). We also derive the average delay bound for QPS-1, which we show is *order-optimal* (*i.e.*, independent of switch size $N$) when the arrival process $A(t)$ further satisfies that for any $i$, $j$, $a_{ij}(0)$ has finite variance. In the sequel, we drop the subscript term from $\rho_\Lambda$ and simply denote it as $\rho$.

Similar to $Q^\dagger_{ij}(t)$, we define $A^\dagger_{ij}(t)$ as the total number of packet arrivals to all VOQs in the neighborhood set $Q^\dagger_{ij}$:

$$A^\dagger_{ij}(t) \triangleq A_{i*}(t) - a_{ij}(t) + A_{*j}(t), \tag{4}$$

where $A_{i*}(t)$ and $A_{*j}(t)$ are similarly defined as $Q_{i*}(t)$ and $Q_{*j}(t)$ respectively. $D^\dagger_{ij}(t)$, $D_{i*}(t)$, and $D_{*j}(t)$ are similarly defined, so is $\Lambda^\dagger_{ij}(t)$. We now state some simple facts concerning $D(t)$, $A(t)$, and $\Lambda$ as follows.

**Fact 1.** *Given any switching algorithm, for any $i$, $j$, we have, $D_{i*}(t) \le 1$ (at most one packet can depart from input port $i$ during time slot $t$), $D_{*j}(t) \le 1$, and $D^\dagger_{ij}(t) \le 2$.*

**Fact 2.** *Given any i.i.d. arrival process $A(t)$ and its rate matrix is $\Lambda$ whose maximum load factor is defined in (3), for any $i$, $j$, we have $\mathbf{E}[A^\dagger_{ij}(t)] = \Lambda^\dagger_{ij} \le 2\rho$.*

The following fact is slightly less obvious.

**Fact 3.** *Given any switching algorithm, for any $i$, $j$, we have*

$$d_{ij}(t)D^\dagger_{ij}(t) = d_{ij}(t). \tag{5}$$

Fact 3 holds because, as mentioned earlier, no other VOQ in $Q^\dagger_{ij}$ (see Figure 1) can be active simultaneously with $q_{ij}$. More precisely, if $d_{ij}(t) = 1$ (*i.e.*, VOQ $q_{ij}$ is active during time slot $t$) then $D^\dagger_{ij}(t) \triangleq D_{i*}(t) - d_{ij}(t) + D_{*j}(t) = 1 - 1 + 1 = 1$; otherwise $d_{ij}(t)D^\dagger_{ij}(t) = 0 \cdot D^\dagger_{ij}(t) = 0 = d_{ij}(t)$.

## 4.2 Why QPS-1 Is Just as Good?

The provable throughput and delay bounds of maximal matching algorithms were derived from a "departure inequality" (to be stated and proved next) that all maximal matchings satisfy. This inequality, however, is not in general satisfied by matchings generated by QPS-1. Rather, QPS-1 satisfies a much weaker form of departure inequality (Lemma 1). Fortunately, this much weaker condition is sufficient for proving the same throughput bound and delay bounds, as will be proved in Theorem 1 and Theorem 2 respectively.

**Property 1** (Departure Inequality, stated as Lemma 1 in [17, 18])**.** *If during a time slot $t$, the crossbar schedule is a maximal matching, then each departure process $D^\dagger_{ij}(t)$ satisfies the following inequality*

$$q_{ij}(t)D^\dagger_{ij}(t) \ge q_{ij}(t). \tag{6}$$

Proof. We reproduce the proof of Property 1 with a slightly different approach for this paper to be self-contained. Suppose the contrary is true, *i.e.*, $q_{ij}(t)D^\dagger_{ij}(t) < q_{ij}(t)$. This can only happen when $q_{ij}(t) > 0$ and $D^\dagger_{ij}(t) = 0$. However, $D^\dagger_{ij}(t) = 0$ implies that no nonempty VOQ (edge) in the neighborhood $Q^\dagger_{ij}$ (see Figure 1) is a part of the matching. Then this matching cannot be maximal (a contradiction) since it can be enlarged by the addition of the nonempty VOQ (edge) $q_{ij}$. □

Clearly, the departure inequality (6) above implies the following much weaker form of it:

$$\sum_{i,j} \mathbf{E}\big[q_{ij}(t)D^\dagger_{ij}(t)\big] \ge \sum_{i,j} \mathbf{E}\big[q_{ij}(t)\big]. \tag{7}$$

In the rest of this section, we prove the following lemma:

**Lemma 1.** *The matching generated by QPS-1, during any time slot $t$, satisfies the much weaker "departure inequality" (7).*

Before we prove Lemma 1, we introduce an important definition and state four facts about QPS-1 that will be used later in the proof. In the following, we will run into several innocuous possible $\frac{0}{0}$ situations that all result from queue-proportional sampling, and we consider all of them to be 0.

We define $\alpha_{ij}(t)$ as the probability of the event that the proposal from input port $i$ to output port $j$ is accepted during the accepting phase, conditioned upon the event that input port $i$ did propose to output port $j$ during the proposing phase. With this definition, we have the first fact

$$\mathbf{E}\big[d_{ij}(t) \mid Q(t)\big] = \frac{q_{ij}(t)}{Q_{i*}(t)} \cdot \alpha_{ij}(t), \tag{8}$$

since both sides (note $d_{ij}(t)$ is a 0-1 *r.v.*) are the probability that $i$ proposes to $j$ and this proposal is accepted. Summing over $j$ on both sides, we obtain the second fact

$$\mathbf{E}\big[D_{i*}(t) \mid Q(t)\big] = \sum_j \frac{q_{ij}(t)}{Q_{i*}(t)} \cdot \alpha_{ij}(t). \tag{9}$$

The third fact is that, for any output port $j$,

$$\mathbf{E}\big[D_{*j}(t) \mid Q(t)\big] = 1 - \prod_i \big(1 - \frac{q_{ij}(t)}{Q_{i*}(t)}\big). \tag{10}$$

In this equation, the LHS is the conditional *probability* ($D_{*j}(t)$ is also a 0-1 *r.v.*) that at least one proposal is received and accepted by output port $j$, and the second term on the RHS of (10) is the probability that no input port proposes to output port $j$ (so $j$ receives no proposal). This equation holds since when $j$ receives one or more proposals, it will accept one of them (the one with the longest VOQ).

The fourth fact is that, for any $i$, $j$,

$$\alpha_{ij}(t) \ge \prod_{k\ne i} \big(1 - \frac{q_{kj}(t)}{Q_{k*}(t)}\big). \tag{11}$$

This inequality holds because when input port $i$ proposes to output port $j$, and no other input port does, $j$ has no choice but to accept $i's$ proposal.

## 4.3 Proof of Lemma 1

Now we are ready to prove Lemma 1.

It suffices to show that for any $i$ and $j$, we have

$$\sum_{i,j} \mathbf{E}\big[q_{ij}(t)D^{\dagger}_{ij}(t) \mid Q(t)\big] \geq \sum_{i,j} q_{ij}(t), \tag{12}$$

because with (12), we have

$$\begin{aligned}\sum_{i,j} \mathbf{E}\big[q_{ij}(t)D^{\dagger}_{ij}(t)\big] &= \mathbf{E}\Big[\mathbf{E}\big[\sum_{i,j} q_{ij}(t)D^{\dagger}_{ij}(t) \mid Q(t)\big]\Big] \\ &\geq \mathbf{E}\big[\sum_{i,j} q_{ij}(t)\big] = \sum_{i,j}\mathbf{E}\big[q_{ij}(t)\big].\end{aligned}$$

By the definition of $D^{\dagger}_{ij}(t) \triangleq D_{i*}(t) - d_{ij}(t) + D_{*j}(t)$, we have,

$$\begin{aligned}&\sum_{i,j} \mathbf{E}\big[q_{ij}(t)D^{\dagger}_{ij}(t) \mid Q(t)\big] \\ =&\sum_{i,j} q_{ij}(t)\mathbf{E}\big[D_{i*}(t) \mid Q(t)\big] - \sum_{i,j} q_{ij}(t)\mathbf{E}\big[d_{ij}(t) \mid Q(t)\big] \\ &+\sum_{i,j} q_{ij}(t)\mathbf{E}\big[D_{*j}(t) \mid Q(t)\big].\end{aligned} \tag{13}$$

Focusing on the first term on the RHS of (13) and using (9), we have,

$$\begin{aligned}&\sum_{i,j} q_{ij}(t)\mathbf{E}\big[D_{i*}(t) \mid Q(t)\big] = \sum_{i} Q_{i*}(t)\mathbf{E}\big[D_{i*}(t) \mid Q(t)\big] \\ =&\sum_{i} Q_{i*}(t)\Big(\sum_{j} \frac{q_{ij}(t)}{Q_{i*}(t)} \cdot \alpha_{ij}(t)\Big) = \sum_{i,j} q_{ij}(t)\alpha_{ij}(t).\end{aligned} \tag{14}$$

Focusing the second term on the RHS of (13) and using (8), we have

$$-\sum_{i,j} q_{ij}(t)\mathbf{E}\big[d_{ij}(t) \mid Q(t)\big] = -\sum_{i,j} q_{ij}(t)\alpha_{ij}(t)\frac{q_{ij}(t)}{Q_{i*}(t)}. \tag{15}$$

Hence, the sum of the first two terms in (13) is equal to

$$\begin{aligned}&\sum_{i,j} q_{ij}(t)\alpha_{ij}(t)\Big(1-\frac{q_{ij}(t)}{Q_{i*}(t)}\Big) \\ \geq&\sum_{i,j} q_{ij}(t)\Big(\prod_{k\neq i}\big(1-\frac{q_{kj}(t)}{Q_{k*}(t)}\big)\Big)\big(1-\frac{q_{ij}(t)}{Q_{i*}(t)}\big) \qquad (16)\\ =&\sum_{i,j} q_{ij}(t)\prod_{i}\big(1-\frac{q_{ij}(t)}{Q_{i*}(t)}\big) \\ =&\sum_{i,j} q_{ij}(t)\Big(1-\mathbf{E}\big[D_{*j}(t) \mid Q(t)\big]\Big). \qquad (17)\end{aligned}$$

Note that (16) is due to (11) and (17) is due to (10). We now arrive at (12), when adding the third and last term in (13) to the RHS of (17).

## 4.4 Throughput Analysis

In this section we prove, through Lyapunov stability analysis, the following theorem (*i.e.,* Theorem 1) which states that any switching algorithm that satisfies the weaker departure inequality (7), including QPS-1 as shown in Lemma 1, can attain at least 50% throughput. The same throughput bound was proved in [4], through fluid limit analysis, for maximal matching algorithms using the (stronger) departure inequality (6) which as stated earlier is not in general satisfied by matchings generated by QPS-1.

**Theorem 1.** *Let $\{Q(t)\}_{t=0}^{\infty}$ be the queueing process of a switching system that is a Markov chain. Let the departure process of $\{Q(t)\}_{t=0}^{\infty}$ satisfy the weaker "departure inequality" (7). Then whenever its maximum load factor $\rho < 1/2$, the queueing process is stable in the following sense: (I) The Markov chain $\{Q(t)\}_{t=0}^{\infty}$ is positive recurrent and hence converges to a stationary distribution $\bar{Q}$; (II) The first moment of $\bar{Q}$ is finite.*

Proof. Here we prove only (I), since Theorem 2 that we will shortly prove implies (II). We define the following Lyapunov function of $Q(t)$: $L\big(Q(t)\big) = \sum_{i,j} q_{ij}(t)Q^{\dagger}_{ij}(t)$, where $Q^{\dagger}_{ij}(t)$ is defined earlier in (1). This Lyapunov function was first introduced in [17] for the delay analysis of maximal matching algorithms for wireless networking. By the Foster-Lyapunov stability criterion [9, Proposition 2.1.1], to prove that $\{Q(t)\}_{t=0}^{\infty}$ is positive recurrent, it suffices to show that, there exists a constant $B>0$ such that whenever the total queue (VOQ) length $\|Q(t)\|_1 > B$ (because it is not hard to verify that the complement set of states $\{Q(t) : \|Q(t)\|_1 \leq B\}$ is finite and the drift is bounded whenever $Q(t)$ belongs to this set), we have

$$\mathbf{E}\big[L\big(Q(t+1)\big) - L\big(Q(t)\big) \mid Q(t)\big] \leq -\epsilon, \tag{18}$$

where $\epsilon > 0$ is a constant. It is not hard to check (for more detailed derivations, please refer to [17]),

$$\begin{aligned}&L\big(Q(t+1)\big) - L\big(Q(t)\big) \\ =&2\sum_{i,j} q_{ij}(t)\big(A^{\dagger}_{ij}(t) - D^{\dagger}_{ij}(t)\big) \\ &+\sum_{i,j}\big(a_{ij}(t) - d_{ij}(t)\big)\big(A^{\dagger}_{ij}(t) - D^{\dagger}_{ij}(t)\big).\end{aligned} \tag{19}$$

Hence the drift (LHS of (18)) can be written as

$$\begin{aligned}&\mathbf{E}\big[L\big(Q(t+1)\big) - L\big(Q(t)\big) \mid Q(t)\big] \\ =&\mathbf{E}\Big[2\sum_{i,j} q_{ij}(t)\big(A^{\dagger}_{ij}(t) - D^{\dagger}_{ij}(t)\big) \mid Q(t)\Big] \\ &+\mathbf{E}\Big[\sum_{i,j}\big(a_{ij}(t) - d_{ij}(t)\big)\big(A^{\dagger}_{ij}(t) - D^{\dagger}_{ij}(t)\big) \mid Q(t)\Big].\end{aligned} \tag{20}$$

Now we claim the following two inequalities, which we will prove shortly.

$$\mathbf{E}\big[2\sum_{i,j} q_{ij}(t)\big(A^{\dagger}_{ij}(t) - D^{\dagger}_{ij}(t)\big) \mid Q(t)\big] \leq 2(2\rho-1)\|Q(t)\|_1. \tag{21}$$

$$\mathbf{E}\big[\sum_{i,j}\big(a_{ij}(t) - d_{ij}(t)\big)\big(A^{\dagger}_{ij}(t) - D^{\dagger}_{ij}(t)\big) \mid Q(t)\big] \leq CN^2. \tag{22}$$

With (21) and (22) substituted into (20), we have

$$\mathbf{E}\big[L\big(Q(t+1)\big) - L\big(Q(t)\big) \mid Q(t)\big] \leq 2(2\rho-1)\|Q(t)\|_1 + CN^2.$$

where $C > 0$ is a constant. Since $\rho < 1/2$, we have $2\rho - 1 < 0$. Hence, there exist $B, \epsilon > 0$ such that, whenever $\|Q(t)\|_1 > B$,

$$\mathbf{E}\big[L\big(Q(t+1)\big) - L\big(Q(t)\big) \mid Q(t)\big] \le -\epsilon.$$

Now we proceed to prove (21).

$$\begin{aligned}
&\mathbf{E}\Big[2\sum_{i,j} q_{ij}(t)\big(A^\dagger_{ij}(t) - D^\dagger_{ij}(t)\big) \mid Q(t)\Big] \\
=&2\Big(\sum_{i,j}\mathbf{E}\Big[q_{ij}(t)A^\dagger_{ij}(t) \mid Q(t)\Big] - \sum_{i,j}\mathbf{E}\Big[q_{ij}(t)D^\dagger_{ij}(t) \mid Q(t)\Big]\Big) \\
\le&2\Big(2\rho\sum_{i,j}\mathbf{E}\Big[q_{ij}(t) \mid Q(t)\Big] - \sum_{i,j}\mathbf{E}[q_{ij}(t) \mid Q(t)]\Big) \qquad (23)\\
=&2(2\rho-1)\|Q(t)\|_1. \qquad (24)
\end{aligned}$$

In the above derivations, inequality (23) holds due to (12), $A(t)$ being independent of $Q(t)$ for any $t$, and Fact 2 that $\mathbf{E}[A^\dagger_{ij}(t)] \le 2\rho$.

Now we proceed to prove (22), which upper-bounds the conditional expectation $\mathbf{E}\big[\big(a_{ij}(t) - d_{ij}(t)\big)\big(A^\dagger_{ij}(t) - D^\dagger_{ij}(t)\big) \mid Q(t)\big]$. It suffices however to upper-bound the unconditional expectation $\mathbf{E}\big[\big(a_{ij}(t) - d_{ij}(t)\big)\big(A^\dagger_{ij}(t) - D^\dagger_{ij}(t)\big)\big]$, which we will do in the following, since we can obtain the same upper bounds on $\mathbf{E}[D^\dagger_{ij}(t)]$ and $\mathbf{E}[d_{ij}(t)]$ (2 and 1 respectively) whether the expectations are conditional (on $Q(t)$) or not. Note the other two terms $A^\dagger_{ij}(t)$ and $a_{ij}(t)$ are independent of (the condition) $Q(t)$.

As for any $i, j$, $a_{ij}(t)$ is *i.i.d.*, we have,

$$\begin{aligned}
&\mathbf{E}\big[\big(a_{ij}(t) - d_{ij}(t)\big)\big(A^\dagger_{ij}(t) - D^\dagger_{ij}(t)\big)\big] \qquad (25)\\
=&\mathbf{E}[a_{ij}(t)A^\dagger_{ij}(t) - d_{ij}(t)A^\dagger_{ij}(t) - a_{ij}(t)D^\dagger_{ij}(t) + d_{ij}(t)D^\dagger_{ij}(t)] \\
=&\mathbf{E}[a^2_{ij}(t)] - \lambda^2_{ij} + \lambda_{ij}\Lambda^\dagger_{ij} - \mathbf{E}[d_{ij}(t)]\Lambda^\dagger_{ij} \\
&- \lambda_{ij}(t)\mathbf{E}[D^\dagger_{ij}(t)] + \mathbf{E}[d_{ij}(t)D^\dagger_{ij}(t)] \\
=&\mathbf{E}[a^2_{ij}(t)] - \lambda^2_{ij} + \lambda_{ij}\Lambda^\dagger_{ij} - \mathbf{E}[d_{ij}(t)]\Lambda^\dagger_{ij} \\
&- \lambda_{ij}(t)\mathbf{E}[D^\dagger_{ij}(t)] + \mathbf{E}[d_{ij}(t)]. \qquad (26)
\end{aligned}$$

In arriving at (26), we have used (2). The RHS of (26) can be bounded by a constant $C > 0$ due to the following assumptions and facts: $\mathbf{E}[a^2_{ij}(t)] = \mathbf{E}[a^2_{ij}(0)] < \infty$ for any $t$, $d_{ij}(t) \le 1$, $D^\dagger_{ij}(t) \le 2$, $\lambda_{ij} \le \rho < 1/2$, and $\Lambda^\dagger_{ij} \le 2\rho < 1$. Therefore, we have (by applying $\sum_{i,j}$ to both (25) and the RHS of (26))

$$\sum_{i,j}\mathbf{E}\Big[\big(a_{ij}(t) - d_{ij}(t)\big)\big(A^\dagger_{ij}(t) - D^\dagger_{ij}(t)\big)\Big] \le CN^2.$$

□

**Remarks.** Now that we have proved that $\{Q(t)\}_{t=0}^\infty$ is positive recurrent. Therefore, for any $i, j$, the long term departure rate $\lim_{T\to\infty}\frac{1}{T}\sum_{t=0}^{T-1}\mathbf{E}[d_{ij}(t)] = \lambda_{ij}$. Hence, we have,

$$\begin{aligned}
&\lim_{T\to\infty}\frac{1}{T}\sum_{t=0}^{T-1}\mathbf{E}\big[\big(a_{ij}(t) - d_{ij}(t)\big)\big(A^\dagger_{ij}(t) - D^\dagger_{ij}(t)\big)\big] \\
=&\sigma^2_{ij} - \lambda_{ij}\Lambda^\dagger_{ij} + \lambda_{ij}. \qquad (27)
\end{aligned}$$

where $\sigma^2_{ij} = \mathbf{E}[a^2_{ij}(t)] - \lambda^2_{ij}$ is the variance of $a_{ij}(t)$, because LHS of (27) is the long term average of (25), and the long term average of (26) can be simplified as the RHS of (27).

Now, we prove the following corollary of Theorem 1 which, in combination with Lemma 1, shows that QPS-1 can attain at least 50% throughput.

**Corollary 1.** *Under an i.i.d. arrival process, whenever the maximum load factor $\rho < 1/2$, QPS-1 is stable in the following sense: The resulting queueing process $\{Q(t)\}_{t=0}^\infty$ is a positive recurrent Markov chain and its stationary distribution $\bar{Q}$ has finite first moment.*

Proof. $\{Q(t)\}_{t=0}^\infty$ is clearly a Markov chain, since in (2), the term $d_{ij}(t)$ is a function of $Q(t)$ and $a_{ij}(t)$ is a random variable independent of $Q(t)$. The rest follows from Lemma 1 and Theorem 1. □

## 4.5 Delay Analysis

In this section, we derive the bound on the expected total queue length $\mathbf{E}[\|\bar{Q}\|_1]$ (readily convertible to the corresponding delay bound using Little's Law) for QPS-1 under *i.i.d.* traffic arrivals using the following moment bound lemma (*i.e.,* Lemma 2) [9, Proposition 2.1.4]. This bound, shown in (29), is identical to that derived in [17, 18, Section III.B] for maximal matchings under *i.i.d.* traffic arrivals. Note this equivalence is not limited to *i.i.d.* traffic arrivals: It can be shown that the delay analysis results for general Markovian arrivals derived in [17, 18] for maximal matchings (using the stronger "departure inequality" (6)) hold also for QPS-1.

**Lemma 2.** *Suppose that $\{Y_t\}_{t=0}^\infty$ is a positive recurrent Markov chain with countable state space $\mathcal{Y}$. Suppose $V$, $f$, and $g$ are non-negative functions on $\mathcal{Y}$ such that,*

$$V(Y_{t+1}) - V(Y_t) \le -f(Y_t) + g(Y_t), \text{ for all } Y_t \in \mathcal{Y}. \qquad (28)$$

*Then $\mathbf{E}[f(\bar{Y})] \le \mathbf{E}[g(\bar{Y})]$, where $\bar{Y}$ is a random variable with the stationary distribution of the Markov chain $\{Y_t\}_{t=0}^\infty$.*

Now we derive the following bound on $\mathbf{E}[\|\bar{Q}\|_1]$, which is stronger than the part (II) of Theorem 1.

**Theorem 2.** *Under the same assumptions and definitions as in Theorem 1, we have*

$$\mathbf{E}[\|\bar{Q}\|_1] \le \frac{1}{2(1-2\rho)}\sum_{i,j}\big(\sigma^2_{ij} - \lambda_{ij}\Lambda^\dagger_{ij} + \lambda_{ij}\big). \qquad (29)$$

Proof. We define $V$, $Y_t$, $f$, and $g$ terms in Lemma 2 in such a way that the LHS and the RHS of (28) become the LHS and the RHS of (19) respectively (e.g., define $V$ as $L$, $Y_t$ as $Q(t)$, and $f(Y_t)$ as $-2\sum_{i,j} q_{ij}(t)\big(A^\dagger_{ij}(t) - D^\dagger_{ij}(t)\big)$). Then, we have,

$$\begin{aligned}
&-2(2\rho-1)\mathbf{E}[\|\bar{Q}\|_1] \\
\le&\mathbf{E}[f(\bar{Y})] \qquad (30)\\
\le&\mathbf{E}[g(\bar{Y})] \qquad (31)\\
=&\sum_{i,j}\big(\sigma^2_{ij} - \lambda_{ij}\Lambda^\dagger_{ij} + \lambda_{ij}\big). \qquad (32)
\end{aligned}$$

In the above derivation, inequality (30) is due to (21) (whose LHS is $-f(Y_t)$), inequality (31) is due to Lemma 2, and equality (32) is due to (27).

Therefore, we have, in steady state,

$$\mathbf{E}[\|\bar{Q}\|_1] \le \frac{1}{2(1-2\rho)}\sum_{i,j}\big(\sigma^2_{ij} - \lambda_{ij}\Lambda^\dagger_{ij} + \lambda_{ij}\big).$$

□

Since, as explained in the proof of Corollary 1, $\{Q(t)\}_{t=0}^{\infty}$ is a Markov chain under i.i.d. arrivals, Theorem 2 applies to QPS-1. Hence we obtain,

**Corollary 2.** *The bound on* $\mathbf{E}[\|\bar{Q}\|_1]$ *as stated in (29) holds under QPS-1 scheduling, whenever the arrival process is i.i.d. and the maximum load factor* $\rho < 1/2$.

It is not hard to check (by applying Little's Law) that the average delay (experienced by packets) is bounded by a constant independent of $N$ (*i.e., order-optimal*) for a given maximum load factor $\rho < 1/2$, if the variance $\sigma_{ij}^2$ for any $i, j$ is assumed to be finite. For the special case of Bernoulli *i.i.d.* arrival (when $\sigma_{ij}^2 = \lambda_{ij} - \lambda_{ij}^2$), this bound (the RHS) can be further tightened to $\frac{\sum_{i,j} \lambda_{ij}}{1-2\rho}$. This implies, by Little's Law, the following "clean" bound: $\bar{\omega} \leq \frac{1}{1-2\rho}$ where $\bar{\omega}$ is the expected delay averaged over all packets transmitting through the switch.

# 5 EVALUATION

In this section, we evaluate, through simulations, the performance of QPS-r under various load conditions and traffic patterns. The main purpose of this section to show that QPS-r performs as well as maximal matching algorithms empirically no just in theory. Hence, we do not compare QPS-r with the two recent iterative algorithms in switching [1]: RR/LQF (Round Robin combined with Longest Queue First) [11] and HRF (Highest Rank First) [10]. Instead, we compare its performance with that of iSLIP [15], a refined and optimized representative parallel maximal matching algorithm (adapted for switching). The performance of the MWM (Maximum Weighted Matching) is also included in the comparison as a benchmark. Our simulations show conclusively that QPS-1 (running 1 iteration) performs very well inside the provable stability region (more precisely, with no more than 50% offered load), and that QPS-3 (running 3 iterations) has comparable throughput and delay performances as iSLIP (running $\log_2 N$ iterations), which has a much higher per-port computational complexity of $O(\log^2 N)$.

## 5.1 Simulation Setup

In our simulations, we fix the number of input/output ports, $N$ to 64. To measure throughput and delay accurately, we assume each VOQ has an infinite buffer size and hence there is no packet drop at any input port. Each simulation run is guided by the following stopping rule [5, 7]: The number of time slots simulated is the larger between $500N^2$ and that is needed for the difference between the estimated and the actual average delays to be within 0.01 with probability at least 0.98.

We assume in our simulations that each traffic arrival matrix $A(t)$ is Bernoulli *i.i.d.* with its traffic rate matrix $\Lambda$ being equal to the product of the offered load $\rho$ and a traffic pattern matrix (defined next). Similar Bernoulli arrivals were studied in [6, 8, 15]. Note that only synthetic traffic (instead of that derived from packet traces) is used in our simulations because, to the best of our knowledge, there is no meaningful way to combine packet traces into switch-wide traffic workloads. The following four standard types of normalized (with each row or column sum equal to 1) traffic patterns are used: (I) *Uniform*: packets arriving at any input port go to each output port with probability $\frac{1}{N}$. (II) *Quasi-diagonal*: packets arriving at input port $i$ go to output port $j=i$ with probability $\frac{1}{2}$ and go to any other output port with probability $\frac{1}{2(N-1)}$. (III) *Log-diagonal*: packets arriving at input port $i$ go to output port $j = i$ with probability $\frac{2^{(N-1)}}{2^N-1}$ and go to any other output port $j$ with probability equal $\frac{1}{2}$ of the probability of output port $j-1$ (note: output port 0 equals output port $N$). (IV) *Diagonal*: packets arriving at input port $i$ go to output port $j = i$ with probability $\frac{2}{3}$, or go to output port $(i \bmod N) + 1$ with probability $\frac{1}{3}$. These traffic patterns are listed in order of how skewed the volumes of traffic arrivals to different output ports are: from uniform being the least skewed, to diagonal being the most skewed.

## 5.2 Throughput and Delay Performances

We first compare the throughput and delay performances of QPS-1 (1 iteration), QPS-3 (3 iterations), iSLIP ($\log_2 64 = 6$ iterations), and MWM (length of VOQ as the weight measure). Figure 2 shows their mean delays (in number of time slots) under the aforementioned four traffic patterns respectively. Each subfigure shows how the mean delay (on a *log scale* along the y-axis) varies with the offered load $\rho$ (along the x-axis). We make three observations from Figure 2. First, Figure 2 clearly shows that, when the offered load is no larger than 0.5, QPS-1 has low average delays (*i.e.*, more than just being stable) that are close to those of iSLIP and MWM, under all four traffic patterns. Second, the maximum sustainable throughputs (where the delays start to "go through the roof" in the subfigures) of QPS-1 are roughly 0.634, 0.645, 0.681, and 0.751 respectively, under the four traffic patterns respectively; they are all comfortably larger than the 50% provable lower bound. Third, the throughput and delay performances of QPS-3 and iSLIP are comparable: The former has slightly better delay performances than the latter under all four traffic patterns except the uniform.

# 6 RELATED WORK

Scheduling in crossbar switches is a well-studied problem with a large amount of literature. So, in this section, we provide only a brief survey of prior work that is directly related to ours, focusing on those we have not described earlier.

**Iterative algorithms that compute maximal matchings.** As mentioned earlier, maximal matchings have long been recognized as a cost-effective family in switching. Among various types of algorithms that compute maximal matchings, the family of parallel iterative algorithms [10, 11, 14, 16, 19, 23] is widely adopted. Parallel iterative algorithms compute a maximal matching via multiple iterations of message exchanges between the input and output ports. Generally, each iteration contains three stages: request, grant, and accept. In the request stage, each input port sends requests to output ports. In the grant stage, each output port, upon receiving requests from multiple input ports, grants to one. Finally, in the accept stage, each input port, upon receiving grants from multiple

[1] Note that they are shown to have reasonably good empirical throughput and delay performance over round-robin-friendly workloads such as uniform and hot-spot traffic when running 1 or 2 iterations. However, as described in §6, they need to run up to $N$ iterations to provably attain at least 50% throughput.

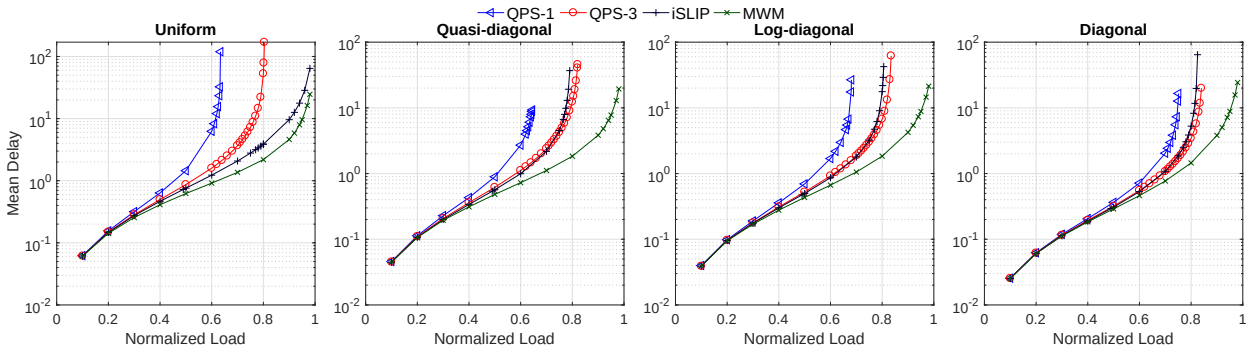


**Figure 2: Mean delays of QPS-1, QPS-3, iSLIP, and MWM under the 4 traffic load patterns.**

output ports, accepts one. Unfortunately, all these parallel iterative algorithms in switching require up to $N$ iterations to guarantee that the resulting matching is a maximal matching. In other words, they need up to $N$ iterations to achieve the same provable throughput and delay performance guarantees as QPS-1 (running 1 iteration).

**Other algorithms that have performance guarantees.** Several serial randomized algorithms, starting with TASS [21] and culminating in SERENA [6], have been proposed that have a total computational complexity of only $O(N)$ yet can provably attain 100% throughput; SERENA, the best among them, also delivers a good empirical delay performance. However, this $O(N)$ complexity is still too high for scheduling high-line-rate high-radix switches, and none of them has been successfully parallelized (*i.e.*, converted to a parallel iterative algorithm) yet.

In [22], a crossbar scheduling algorithm specialized for switching variable-size packets was proposed, that has $O(1)$ total computational complexity. Although this algorithm can provably attain 100% throughput, its delay performance is poor. For example, as shown in [8], its average delays, under the aforementioned four standard traffic matrices, are roughly 3 orders of magnitudes higher than those of SERENA [6] even under a moderate offered load of 0.6.

## 7 CONCLUSION

In this work, we propose QPS-r, a parallel iterative switching algorithm with $O(1)$ computational complexity per port. We prove, through Lyapunov stability analysis, that it achieves the same throughput and delay performance guarantees in theory, and demonstrate through simulations that it has comparable performances in practice as the family of maximal matching algorithms (adapted for switching); maximal matching algorithms are much more expensive computationally (at least $O(\log N)$ iterations and a total of $O(\log^2 N)$ per-port computational complexity). These salient properties make QPS-r an excellent candidate algorithm that is fast enough computationally and can deliver acceptable throughput and delay performances for high-link-rate high-radix switches.

**Acknowledgments**. This work is supported in part by US NSF through award CNS 1909048 and CCF 1850439.

## REFERENCES

[1] Thomas E. Anderson, Susan S. Owicki, James B. Saxe, and Charles P. Thacker. 1993. High-speed Switch Scheduling for Local-area Networks. *ACM Trans. Comput. Syst.* 11, 4 (Nov. 1993), 319–352.

[2] Cagla Cakir, Ron Ho, Jon Lexau, and Ken Mai. 2015. Modeling and Design of High-Radix On-Chip Crossbar Switches. In *Proc. of the ACM/IEEE NoCS*. Vancouver, BC, Canada, Article 20, 8 pages.

[3] Cagla Cakir, Ron Ho, Jon Lexau, and Ken Mai. 2016. Scalable High-Radix Modular Crossbar Switches. In *Proceedings of the HOTI*. Santa Clara, CA, USA, 37–44.

[4] Jim Dai and Balaji Prabhakar. 2000. The Throughput of Data Switches with and without Speedup. In *Proceedings of the IEEE INFOCOM*. Tel Aviv, Israel, 556–564.

[5] James M Flegal, Galin L Jones, et al. 2010. Batch means and spectral variance estimators in Markov chain Monte Carlo. *Ann. Stat.* 38, 2 (2010), 1034–1070.

[6] Paolo Giaccone, Balaji Prabhakar, and Devavrat Shah. 2003. Randomized Scheduling Algorithms for High-Aggregate Bandwidth Switches. *IEEE J. Sel. Areas Commun.* 21, 4 (May 2003), 546–559.

[7] Peter W Glynn, Ward Whitt, et al. 1992. The Asymptotic Validity of Sequential Stopping Rules for Stochastic Simulations. *Ann. Appl. Probab.* 2, 1 (1992), 180–198.

[8] Long Gong, Paul Tune, Liang Liu, Sen Yang, and Jun (Jim) Xu. 2017. Queue-Proportional Sampling: A Better Approach to Crossbar Scheduling for Input-Queued Switches. *Proceedings of the ACM SIGMETRICS* 1, 1 (June 2017), 3:1–3:33.

[9] Bruce Hajek. 2006. Notes for ECE 467 Communication Network Analysis. http://www.ifp.illinois.edu/~hajek/Papers/networkanalysisDec06.pdf.

[10] B. Hu, F. Fan, K. L. Yeung, and S. Jamin. 2018. Highest Rank First: A New Class of Single-Iteration Scheduling Algorithms for Input-Queued Switches. *IEEE Access* 6 (2018), 11046–11062.

[11] B. Hu, K. L. Yeung, Q. Zhou, and C. He. 2016. On Iterative Scheduling for Input-Queued Switches With a Speedup of $2 - 1/N$. *IEEE/ACM Trans. Netw.* 24, 6 (December 2016), 3565–3577.

[12] Amos Israel and A. Itai. 1986. A Fast and Simple Randomized Parallel Algorithm for Maximal Matching. *Inf. Process. Lett.* 22, 2 (Feb. 1986), 77–80.

[13] M. Karol, M. Hluchyj, and S. Morgan. 1987. Input Versus Output Queueing on a Space-Division Packet Switch. *IEEE Trans. Commun.* 35, 12 (December 1987), 1347–1356.

[14] D. Lin, Y. Jiang, and M. Hamdi. 2011. Selective-Request Round-Robin Scheduling for VOQ Packet Switch Architecture. In *Proceedings of the IEEE ICC*. 1–5.

[15] Nick McKeown. 1999. The iSLIP Scheduling Algorithm for Input-queued Switches. *IEEE/ACM Trans. Netw.* 7, 2 (Apr. 1999), 188–201.

[16] Nicholas William McKeown. 1995. *Scheduling Algorithms for Input-queued Cell Switches*. Ph.D. Dissertation. Berkeley, CA, USA. UMI Order No. GAX96-02658.

[17] M. J. Neely. 2008. Delay Analysis for Maximal Scheduling in Wireless Networks with Bursty Traffic. In *Proceedings of the IEEE INFOCOM*.

[18] M. J. Neely. 2009. Delay Analysis for Maximal Scheduling With Flow Control in Wireless Networks With Bursty Traffic. *IEEE/ACM Trans. Netw.* 17, 4 (Aug 2009), 1146–1159.

[19] A. Scicchitano, A. Bianco, P. Giaccone, E. Leonardi, and E. Schiattarella. 2007. Distributed Scheduling in Input Queued Switches. In *Proceedings of the IEEE ICC*. 6330–6335.

[20] Y. Tamir and G. L. Frazier. 1988. High-performance Multi-queue Buffers for VLSI Communications Switches. *SIGARCH Comput. Archit. News* 16, 2 (May 1988), 343–354.

[21] Leandros Tassiulas. 1998. Linear Complexity Algorithms for Maximum Throughput in Radio Networks and Input Queued Switches. In *Proceedings of the IEEE INFOCOM*. San Francisco, CA, USA, 533–539.

[22] Shunyuan Ye, Tanming Shen, and Shivendra Panwar. 2010. An $O(1)$ Scheduling Algorithm for Variable-Size Packet Switching Systems. In *Proceedings of the 48th Annual Allerton Conference*. 1683–1690.

[23] Yihan Li, S. Panwar, and H. J. Chao. 2001. On The Performance of A Dual Round-Robin Switch. In *Proceedings of the IEEE INFOCOM*. Anchorage, AK, USA, 1688–1697 vol. 3.