An algorithm for improved delay-scaling in input-queued switches

An algorithm for improved delay-scaling in input-queued switches

Noname manuscript No.
(will be inserted by the editor)

An Algorithm for Improved Delay-Scaling in Input Queued Switches

Wentao Weng · R. Srikant

Received: date / Accepted: date

Abstract We consider an $n \times n$ input-queued switch with uniform Bernoulli traffic and study the delay (or equivalently, the queue length) in the regime where the size of the switch n and the load (denoted by ρ) simultaneously become large. We devise an algorithm with expected total queue length equal to $O((n^{5/4}(1-\rho)^{-1})\log\max(1/\rho, n))$ for large n and ρ such that $(1-\rho)^{-1} \geq n^{3/4}$. This result improves the previous best queue length bound in the regime $n^{3/4} < (1-\rho)^{-1} < n^{7/4}$. Under same conditions, the algorithm has an amortized time complexity $O(n+(1-\rho)^2 n^{7/2}/\log\max(1/\rho, n))$. The time complexity becomes $O(n)$ when $(1-\rho)^{-1} \geq n^{5/4}$.

Keywords Input-queued switch · Queue-size scaling · Stochastic network · Large systems

Mathematics Subject Classification (2010) 68M20 · 60K25 · 60C05 · 90B15

1 Introduction

Input-queued switch models have been quite popular in the context of computer communication networks, originally as models for the switching fabric in Internet switches and more recently as models for data center networks. They are also of independent interest to the applied probability and performance analysis communities as prototypical models of queueing networks with interacting resources. Mathematically, an input-queued switch can be viewed as a

W. Weng
Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, MA, USA
E-mail: wweng@mit.edu

R. Srikant
Coordinated Science Lab, University of Illinois at Urbana-Champaign, Urbanna, IL, USA
E-mail: rsrikant@illinois.edu

matrix of queues operating in discrete-time, with independent packet arrival processes to each queue, which are often taken to be Bernoulli processes for tractability. The key distinguishing property of an input-queued switch is the following service constraint: at each time instant, we are only allowed to serve at most one queue in each row and one queue in each column of the matrix of queues. Additionally, when a queue is served, at most one packet can be removed from it.

The input-queued switch model was first studied to design low-complexity throughput-optimal scheduling algorithms. The scheduling algorithm called the MaxWeight algorithm was presented in [22] where the authors showed that scheduling using a maximum weighted matching, with queue lengths as weights, is throughput optimal. The algorithm was designed for networks that were more general than input-queued switches. The algorithm was rediscovered in the context of input-queued switches in [13] where it was also shown that lower-complexity scheduling algorithms such as maximum size and maximal matchings fail to be throughput optimal. Additionally, in [24] and [6], it was shown that the simpler maximal matching algorithm achieves at least half the maximum throughput region.

Following throughput-optimality results, there has been much interest in designing algorithms which are also delay optimal for input-queued switches. Note that, since we can only serve at most one queue in each row and in each column in each time slot, the total arrival rate to each column and to each row must be less than one packet per time slot to ensure the stability of the queueing network. Let ρ denote the maximum arrival rate to any column or row in the network. It has been conjectured that there exists an algorithm under which the total queue length in the network scales as $O(n/(1-\rho))$. Using Little's law, this statement is equivalent to saying that the delay scales as $O(1/(1-\rho))$. Another variant of the conjecture states that such a scaling holds for the MaxWeight algorithm. These conjectures have been difficult to prove, so a number of variants of the problem have been considered in the literature which we will review next.

It was shown in [14,16] that the total queue length in the switch scales as $O(n)$ under the maximal matching algorithm if $\rho < 0.5$. The result is interesting due to two reasons: (i) it shows that the delay in the switch is independent of the size of the switch at least when the traffic is light and (ii) the result holds for the maximal matching algorithm which has low computational complexity. The other extreme traffic regime is the heavy-traffic regime where n is fixed and $\rho \to 1$. It was shown in [12] that product of the total queue length and $(1-\rho)$ converges to $O(n)$ in the heavy-traffic regime under the MaxWeight algorithm. The result in [12] builds upon a Lyapunov drift method developed in [7] which was further motivated by fluid models of state-space collapse exhibited by input-queued switches [21,2,19] operating under the MaxWeight algorithm. A common feature of these results is that the underlying algorithm does not require any knowledge of ρ and the results hold for non-uniform and non-Bernoulli traffic.

Since the above results do not fully address the $O(n/(1-\rho))$ queue length-scaling conjecture, other relaxations have been considered for the scaling regime, the traffic model and the scheduling algorithms. In [15], it was shown that an algorithm which assumes an upper bound on ρ and uses a batch-scheduling model can achieve a queue length of $O((n/(1-\rho)^2)\log n)$. The work [18] proposed an algorithm with exponential running time that has a queue length of $O(n/(1-\rho)+n^3)$. If $1-\rho=1/n$, then the queue length scales as $O(n^3)$ under both algorithms in [15,18] if neglecting logarithm terms. This motivates the question of whether it is possible to get a smaller queue length in this traffic regime. This question has been answered for the special case of uniform Bernoulli traffic in [17] who showed a scaling of $O((n^{3/2}/(1-\rho))\log(1-\rho)^{-1})$, which was further improved in [25] to $O((n/(1-\rho)^{4/3})\log\max(n,(1-\rho)^{-1}))$. It should be noted that the algorithms in [17,25] are batch-scheduling algorithms of the type introduced in [15] but modified further to improve delay performance. Additionally, the result in [25] does not require $\rho=1-1/n$ but does assume uniform, Bernoulli traffic. Our goal in this paper is to improve the best known scaling result under uniform, Bernoulli traffic in the scaling regime $\rho=1-1/n$ or higher.

With this goal in mind, our main contribution in the paper is the following: we design a new algorithm for scheduling in input-queue switches that which leads to an expected total queue length equal of $O((n^{5/4}(1-\rho)^{-1})\log\max(1/\rho,n))$ for ρ such that $(1-\rho)^{-1}\geq n^{3/4}$. The new result improves the previously known queue length bound in the regime $n^{3/4}<(1-\rho)^{-1}<n^{7/4}$. Our algorithm is of batching type, and builds upon an integration of the Round-Robin idea in [17] and the lower envelope idea in [25]. Crucially, the new algorithm utilizes the fact that Round-Robin allows each queue to be served for an equal fraction of time. Such a property enables successive implementation of Round-Robin and lower envelope matching, leading to a better scheduling algorithm. Benefiting from the batching schedule, the proposed algorithm has a small amortized time complexity $O(n+(1-\rho)^2n^{7/2}/\log\max(1/\rho,n))$. When $(1-\rho)^{-1}\geq n^{5/4}$, the complexity becomes $O(n)$, matching the optimal running time of other low-complexity algorithms [11].

Notation: For two matrices A and B, we use $A\geq B$ to denote that every element of A is greater than or equal to the corresponding element of B. We denote the set $\{1,2,\cdots,n\}$ as $[n]$ for $n\in\mathbb{Z}^+$. This paper uses asymptotic notations. Let x be a positive parameter, and $f(x),g(x)$ be two positive real-valued function. We write $f(x)=O(g(x))$ if $\limsup_{x\to\infty}\frac{f(x)}{g(x)}<\infty$; $f(x)=\omega(g(x))$ if $\liminf_{x\to\infty}\frac{f(x)}{g(x)}=\infty$.

2 Model

Consider an $n\times n$ input-queued switch where there are n input ports and n output ports, and ports are labelled from 1 to n on each side. For each pair of input port i and output port j, there is a queue Q_{ij} which stores packets that arrive at input port i which need to be routed to output port j. We assume time is slotted and takes values in $\{1, 2, \cdots\}$. Every queue is empty at slot 1. At a time slot τ, the following events happen sequentially.

First, the scheduler decides a schedule $\sigma = (\sigma_{ij})_{n\times n} \in \{0,1\}^{n\times n}$. If $\sigma_{ij} = 1$, a packet in queue Q_{ij} is scheduled. But if there is no packet in the queue, the service is wasted. Due to the internal structure of a switch, the schedule can include only one packet out of an input port, and only one packet into an output port. Mathematically, the constraint can be written as

$$\forall_i, \sum_{j'=1}^{n} \sigma_{ij'} \leq 1;\ \forall_j, \sum_{i'=1}^{n} \sigma_{i'j} \leq 1. \tag{1}$$

In other words, the schedule σ must be a matching. We may interchangeably use the terms matchings or schedules for convenience. Let $\mathcal{S}$ be the set of perfect matchings, i.e.,

$$\forall_i, \sum_{j'=1}^{n} \sigma_{ij'} = 1;\ \forall_j, \sum_{i'=1}^{n} \sigma_{i'j} = 1. \tag{2}$$

As we can see, if we denote σ_i to be the unique j such that $\sigma_{ij} = 1$, then σ is exactly a permutation of length n. On the other hand, if a queue matrix $\mathbf{q}$ satisfying $\mathbf{q} \geq \sigma$ for a perfect matching σ, then this schedule can serve exactly n packets, which is the maximum amount of service per slot.

After scheduled packets leave, new packets arrive into the system. For each queue Q_{ij}, we assume that a new packet arrives with probability $\frac{\rho}{n}$, and no arrival arrives with probability $1 - \frac{\rho}{n}$. We assume that packet arrivals are independent among different queues. That is to say, the arrival of packets to each queue follows an independent Bernoulli process with rate $\frac{\rho}{n}$. It is assumed that $\rho \in (0,1)$, and ρ can scale with n. The load of the system is also ρ because for each port, the arrival rate is exactly ρ, and the amount of service per slot is at most 1.

With a little abuse of notation, we use $Q_{ij}(\tau)$ to denote the queue length of queue Q_{ij} at the beginning of slot τ. Similarly, define $A_{ij}(\tau)$ to be the total number of arrivals to Q_{ij}, and $S_{ij}(\tau)$ to be the total number of packets scheduled in Q_{ij} during the first τ slots. We assume $A_{ij}(0) = S_{ij}(0) = 0$. Let $\sigma(\tau)$ be the chosen schedule for slot τ. Then the queue matrix $\mathbf{Q}(\tau) = (Q_{ij}(\tau))_{n\times n}$ evolves as

$$\begin{aligned} Q_{ij}(\tau+1) &= (Q_{ij}(\tau) - \sigma_{ij}(\tau))^+ + A_{ij}(\tau) - A_{ij}(\tau-1) \\ Q_{ij}(\tau+1) &= A_{ij}(\tau) - S_{ij}(\tau). \end{aligned} \tag{3}$$

The main concern is this paper is to find a policy that can minimize the expected queue length $\mathbb{E}\left[\sum_{ij} Q_{ij}(\tau)\right]$ at every time slot τ.

3 Main Result

In this section, we present the main result of this paper. Define $f = \max(n, (1-\rho)^{-1})$. Since we are considering a dynamical system, the amortized time complexity of an algorithm is defined as the time-average computation, given by $\lim_{t\to\infty} \frac{\sum_{i=1}^{t} T(i)}{t}$ where $T(i)$ is the computation time at slot i.

Theorem 1 *Consider an $n \times n$ switch with independent Bernoulli arrival processes to each queue of rate $\frac{\rho}{n}$. Suppose $(1-\rho)^{-1} \geq n^{0.75}$. Then there exist a scheduling policy such that*

- *the amortized time complexity is $O(n + (1-\rho)^2 n^{3.5}/\log(f))$.*
- *for every time slot τ, it holds that for large enough n,*

$$\mathbb{E}\left[\sum_{i,j} Q_{ij}(\tau)\right] \leq 3n^{1.25}(1-\rho)^{-1}\log(f).$$

We would like to highlight that the policy claimed in Theorem 1 can be constructed, and its construction is postponed until Section 5. When the load ρ scales as $1-\frac{1}{n}$, we have the following bound on the expected queue length. This new bound $O(n^{9/4}\log n)$ improves the previous best known result $O(n^{7/3}\log n)$ in [25].

Corollary 1 *Under the setting in Theorem 1, suppose $\rho = 1 - \frac{1}{n}$. Then there exists a scheduling policy with amortized time complexity $O(n^{1.5}/\log(n))$ such that for every time slot τ, the following bound holds: for large enough n,*

$$\mathbb{E}\left[\sum_{i,j} Q_{ij}(\tau)\right] \leq 3n^{2.25}\log n.$$

4 Preliminaries

Before we describe our policy, we first introduce two known results which are essential building blocks in the policy.

4.1 Minimum Clearance Time

The first result concerns the minimum clearance time of a queue matrix. Specifically, the goal is to minimize the number of time slots to clear packets from a fixed queue matrix $\mathbf{q} = (q_{ij})_{n\times n}$. We call a policy that has minimum clearance time an optimal clearing policy.

Let γ be the maximum number of packets among all input ports and output ports in $\mathbf{q}$. Then the following theorem states that packets in $\mathbf{q}$ can be cleared in exactly γ slots. The following theorem is adopted from [15, Fact 3].

Theorem 2 *Let* $\mathbf{q} = (q_{ij})_{n\times n}$ *be a queue matrix. For every* $1 \le i \le n, 1 \le j \le n$*, denote*

$$h_i = \sum_{j'=1}^{n} q_{ij'}; c_j = \sum_{i'=1}^{n} q_{i'j},$$

and let $\gamma = \max\{\max_i h_i, \max_j c_j\}$ *. Then there exists a sequence of* γ *matchings* $\sigma^{(1)}, \cdots, \sigma^{(\gamma)}$*, such that* $\sum_{p=1}^{\gamma} \sigma^{(p)} \ge \mathbf{q}$.

As noted in previous work, such a policy can be found using Maximum Size Matching by augmenting a bipartite graph with additional edges to make all node degrees equal [15] or using Node Weighted Matching [10]. However, both algorithms have $O(n^{2.5})$ complexity per time slot, and thus take $O(\gamma n^{2.5})$ time to find the clearing policy, which is not desirable when γ scales with n.

We remark that we can indeed find such policy by finding a minimum edge coloring on a bipartite graph. To see this fact, fix a queue matrix $\mathbf{q} = (q_{ij})_{n\times n}$. We can create a bipartite graph G with n left nodes and n right nodes. If an input port i has q_{ij} packets to output port j, we create q_{ij} edges from the i^{th} left node to the j^{th} right node in G. Let γ denote the maximum number of packets in all ports as defined in Theorem 2. We color each edge using one of γ colors such that any two edges that share a common endpoint have different colors. Then if we view one color as one time slot, the set of edges with the same color forms a feasible matching for the corresponding time slot. We can thus find an optimal clearing policy by edge coloring on a bipartite graph. A similar technique is also used in [1,23].

The best known algorithm for bipartite edge coloring runs in $O(m \log n)$ time [5] where m is the number of edges. Therefore, it can greatly improve the time complexity in contrast to $O(\gamma n^{2.5})$.

4.2 Lower Envelopes

We can see Theorem 2 basically provides an upper bound type of result for a given queue matrix, i.e., how many perfect matchings we need to cover $\mathbf{q}$. Now we describe a lower bound result that characterizes the maximum number of perfect matchings we can find from $\mathbf{q}$. Results in this section are from [25]. We first define $\beta-$lower envelopes of a queue matrix $\mathbf{q}$. Although the definition here is slightly different from that in [25] for better illustration, the equivalence of the two definitions is justified in [25, Remark 1].

Definition 1 A $\beta-$lower envelope of a queue matrix $\mathbf{q}$ is a sequence of β perfect matchings $\sigma^{(1)}, \cdots, \sigma^{(\beta)}$ such that $\mathbf{q} \ge \sum_{p=1}^{\beta} \sigma^{(p)}$.

The existence of a $\beta-$lower envelope is given by following result.

Theorem 3 *[25, Proposition 1] Consider a queue matrix* $\mathbf{q} = (q_{ij})_{n\times n}$*. There exists a* $\beta-$*lower envelope of* $\mathbf{q}$ *if and only if for any* $\mathcal{R}, \mathcal{C} \subseteq [n]$ *with* $|\mathcal{R}| = k$ *and* $|\mathcal{C}| = \ell$*, we have*

$$\beta(k + \ell - n) \le \sum_{i\in\mathcal{R}, j\in\mathcal{C}} q_{ij}. \tag{4}$$

Although the work [25] did not analyze the explicit complexity to find a $\beta-$lower envelope of a queue matrix $\mathbf{q}$, we include one here for completeness.

We first describe how to find a $\beta-$lower envelope. The first step is to verify the existence of a $\beta-$lower envelope of $\mathbf{q}$. As in [25], we construct a directed network graph G with a source s, and a sink t. The graph G has n left nodes, n right nodes and the two nodes s, t. For each left node, there is an edge from s to this node with capacity β. Similarly, for each right node, there is an edge from this node to t with capacity β. Finally, for the i^{th} left node and j^{th} right node, there is an edge e_{ij} from i to j with capacity q_{ij}.

We then find the maximum flow of G. If the maximum flow is not $n\beta$, the queue matrix $\mathbf{q}$ does not have an $\beta-$lower envelope [25]. Otherwise, the flow on each edge e_{ij} constitutes a new matrix $\mathbf{f} = (f_{ij})_{n\times n}$ where the sum of each row and column is exactly β, and $\mathbf{f} \leq \mathbf{q}$. By Theorem 2, we can find β perfect matchings $\sigma^{(1)}, \cdots, \sigma(\beta)$ such that their sum is exactly $\mathbf{f}$.

The time complexity of the above algorithm consists of two parts: to find maximum flow and to find an optimal clearing policy. From the discussion in Section 4.1, there is an $O(\beta n \log n)$ algorithm for the second part because the sum of elements in $\mathbf{f}$, which is the number of edges in the corresponding bipartite graph, is βn. Note that β can depend on n, so it may not be a constant. For the first part, although there are a vast amount of network flow algorithms [9], we consider Goldberg's push-relabel algorithm [8] for its nice time complexity equal to $O(n^3)$, and even $O(n^2 \log n)$ with a parallel implementation. Summing up above discussions, the time complexity to find a $\beta-$lower envelope is $O(n(n^2 + \beta \log n))$.

5 Policy Description

In this section, we describe our policy for the input-queued switch. Our policy is a batching policy along the lines of previous work [15, 17, 25]. We first provide an overview of the policy. Then we provide details on how the core component of the policy, Recursive Clearing, is implemented. Finally, we give explicit parameter settings used in the policy.

5.1 Policy Overview

We assume that time slots are separated into intervals of length b which we call arrival periods, and the arrival period $\{kb+1, \cdots, (k+1)b\}$ is given index k where $k \in \mathbb{Z}^+$. Fix an arrival period k, we will serve packets from this period in its corresponding service period as shown in Figure 1.

Specifically, the policy follows a similar structure as in [17, 25]. However, it uses additional strategies to schedule packets that help reduce the expected queue length.

Define $Q_{ij}^k(t)$ to be the number of packets in queue Q_{ij} at the beginning of time slot $kb+t$ that arrive in arrival period k. Let $\mathbf{Q}^k(t) = (Q_{ij}^k(t))_{n\times n}$ be

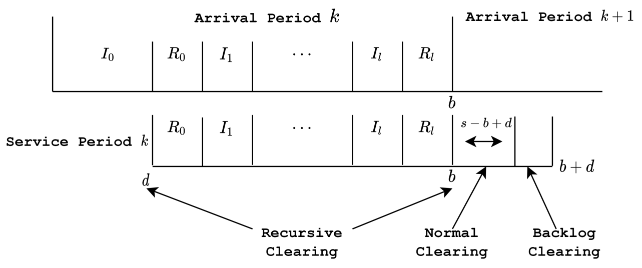

Fig. 1 Policy Overview

the whole queue matrix. Initially, $Q_{ij}^k(1) = 0$ for all i, j. We assume that all arrivals in the current period will directly join $\mathbf{Q}^k$. The policy works as follows in each arrival period. Note that the word "phase" is used for a sub-interval in an arrival period or a service period.

1. First, for a fixed parameter d, no arrival in the first d slots $\{kb+1, \cdots, kb+d\}$ is served. This phase is labelled as I_0 in Figure 1, which we call the **Idling** phase.
2. Then for the next $b-d$ slots, namely $\{kb+d+1, \cdots, (k+1)b\}$, the policy sequentially goes through two kinds of phases, **Round-Robin** and **Packet-Collecting**. Two different algorithms are used in these phases to schedule packets arriving in the current arrival period. These phases are labelled as $R_u, u \geq 0$ for Round-Robin phases, and $I_u, u \geq 1$ for Packet-Collecting phases in Figure 1. Note that I_0 is reserved for the Idling phase. We assume there are in total $l+1$ pairs of (I_u, R_u) where $0 \leq u \leq l$. In phase R_u with $u \geq 0$, the scheduler can schedule all remaining packets in $\mathbf{Q}^k$. But in phase I_u with $u \geq 1$, the scheduler only schedules packets in $\mathbf{Q}^k$ that arrive before I_u. We note that I_0 is indeed a special case because $\mathbf{Q}^k$ is empty at the beginning.
   The whole phase, from R_0 to R_l, is called **Recursive Clearing** because we are recursively scheduling arriving packets. Details of algorithms in Recursive Clearing is given in the next section.
3. The third phase is **Normal Clearing** of length $s-b+d$ where s is a fixed value related to n, ρ. This phase includes slots in $\{(k+1)b+1, \cdots, kb+s+d\}$, and all packets in arrival period k have arrived. As a result, we can use the optimal clearing policy introduced in Theorem 2 to clear remaining packets in $\mathbf{Q}^k$. But to ensure a low time complexity, we first check whether the maximum number of packets among all ports is below the phase length $s-b+d$. If it is, then we evoke the algorithm in Section 4.1 to find the optimal clearing policy. Otherwise, we skip this step, and put all remaining packets into a backlog queue, as shown below.
   Let U_k denote the number of left packets. We maintain a global backlog queue $\mathbf{B}$, and all remaining packets in $\mathbf{Q}^k$ will be moved to the backlog queue. Note that the queue $\mathbf{B}$ can include backlogs from previous arrival periods. Therefore, the total number of packets in all queues at the time

slot τ is given by the sum of queue length B_k, the number of packets in $\mathbf{Q}^k$, and new arrivals in the next period.

4. The final phase is **Backlog Clearing** of length $b-s$ with slots $\{kb+s+d+1,\cdots,(k+1)b+d\}$. We can see the total length of Normal Clearing and Backlog Clearing is exactly d, which is equal to the length of the Idling phase of the next arrival period.
   As the name suggests, in Backlog Clearing, backlogs in queue $\mathbf{B}$ will be scheduled. We assume that at each time slot in this phase, exactly one packet from $\mathbf{B}$ will be scheduled (if any exists). Let B_k denote the number of backlog packets in $\mathbf{B}$ at the beginning of arrival period k. Then we can see that B_k has the following update equation:

$$B_{k+1} = \left(B_k + U_k - (b-s)\right)^+ . \tag{5}$$

We remark that the Idling phase of an arrival period $k+1$ is indeed the same time interval as the Normal Clearing phase and the Backlog Clearing phase of arrival period k. Therefore, the algorithm is trying to schedule for nearly all time slots except the first I_0 time slots of the first arrival period.

5.2 Recursive Clearing

We detail what algorithms are used in Recursive Clearing in this section. Note that we may also denote I_u (or R_u) as the length of the phase I_u (or R_u). The meaning should be clear from the context. Denote $T_u = I_u + R_u$ for all $0 \le u \le l$.

5.2.1 Round-Robin Phase

This phase is motivated by the Round-Robin algorithm in [17]. Fix a Round-Robin phase R_u where $0 \le u \le l$. We will run a Round-Robin policy that has n permutation matrices $\sigma^{(0)},\cdots,\sigma^{(n-1)}$ of size n such that for $\sigma^{(p)}$, it satisfies

$$\sigma^{(p)}(i,j) = \begin{cases} 1, & \text{if } j \equiv (i+p) \pmod n \\ 0, & \text{otherwise.} \end{cases} \tag{6}$$

During R_u, we sequentially use $\sigma^{(0)},\cdots,\sigma^{(n-1)},\sigma^{(0)},\cdots$ as the scheduling policy for $\mathbf{Q}^k(\tau - kb)$ for each time slot τ. Then if a cycle of $\{\sigma^p, p<n\}$ is used, every pair of ports (i,j) will be scheduled exactly once.

5.2.2 Packet-Collecting Phase

In Packet-Collecting, instead of scheduling in a Round-Robin manner, we simply schedule some perfect matchings of a queue matrix as in [25]. To be specific, let us fix a Packet-Collecting phase I_u where $u \ge 1$. Denote $L = \sum_{p=0}^{u-1} T_p$, that is, the ending slot of the last phase. Then in I_u, we only schedule packets in $\mathbf{Q}^k(L+1)$. For any new arrivals during I_u, we will put them in a backup queue. At the end of this phase, we relocate them into the initial queue $\mathbf{Q}^k$. We try to find exactly I_u perfect matchings $\sigma^{(1)}, \cdots, \sigma^{(I_u)}$ such that $\mathbf{Q}^k(L+1) \geq \sum_{p=1}^{I_u} \sigma^{(p)}$. Such a set of perfect matchings is an I_u-lower envelope of the queue matrix $\mathbf{Q}^k(L+1)$ introduced in Section 4.2. Therefore, there exists efficient network flow algorithms that can verify the existence of such a set of matchings, and provide a specific solution if one exists.

The policy first tests whether such an I_u-lower envelope exists. If there isn't a feasible solution, the policy does nothing in this phase. On the other hand, if there exists such a solution $\sigma^{(1)}, \cdots, \sigma^{(I_u)}$, we schedule these perfect matchings one by one for the I_u time slots in this phase.

5.3 Intuition of Phase Length and Delay Improvement

Before diving into details of the algorithm and the performance analysis, let us first provide an intuitive explanation of the queue length bound given in Theorem 1 and how to set the length of each phase.

Roughly speaking, the algorithm ensures that all packets arriving in an arrival period will get service during Recursive clearing and Normal Clearing. Then each arrival period is almost independent and the queue matrix $\mathbf{q}$ is approximately zero at the start of an arrival period. Fix an arrival period. The intuition behind the algorithm is to guarantee that for each time slot $t + I_0$ in Recursive Clearing, the realized schedule is a perfect matching. Then the expected queue length at time slot $t + I_0$ is upper bounded by $n(t + I_0) - nt = nI_0$, which only depends on the length of Idling phase. Then since we use Round-Robin in the service period R_u to serve packets arriving in I_u and R_u, we need about $\frac{R_u}{n} \leq \frac{T_u \rho}{n} - \sqrt{T_u \frac{\rho}{n} \log n}$ to ensure that $Q_{ij}(t + I_0) > 0$ and no packets are wasted. Similarly, since we use Packet-Collecting in I_u to serve all unserved packets by Round-Robin in I_{u-1} and R_{u-1}, we need $I_u \leq \rho T_{u-1} - R_{u-1} - \sqrt{T_{u-1} \rho \log n}$. The final constraint is to ensure all remaining packets can be cleared in Normal Clearing, which requires that $\sum T_u \geq \rho \sum T_u + \sqrt{\rho \sum T_u \log n}$ and gives $\sum T_u \geq (1-\rho)^{-2}$. Consider the important special case where $\rho = 1 - \frac{1}{n}$. The above constraints motivate us to set $I_u \approx n^{1.25}, R_u \approx n^{1.5}$. It then shows that the queue length is about $O(n^{2.25})$, ignoring additional logarithmic factors.

5.4 Parameter Settings

In this section, we provide explicit settings of parameters for the policy, and check that the above policy is well-defined. We assume n is large enough when selecting parameters.

Define $f = \max(n, (1-\rho)^{-1})$. Let $\widetilde{R}$ be the largest multiple of n that is no larger than $2n^{-0.5}(1-\rho)^{-2} \log f$. For all parameters, we assume they are rounded up to integers as it will not affect the result when n is large enough.

Let c_d be a positive constant, and denote $c_p = \left\lfloor \frac{c_d}{16\sqrt{19(c_d+2)}} \right\rfloor$. We set

$$\begin{aligned} I_0 = d = c_d n^{0.25}(1-\rho)^{-1}\log f \\ R_0 = R_1 = \cdots = R_l = \widetilde{R},\ l = c_p n^{0.5}, \end{aligned} \tag{7}$$

and for each u with $1 \le u \le l$,

$$I_u = d - u \cdot \left(8\sqrt{19(c_d+2)}\right) n^{-0.25}(1-\rho)^{-1}\log f. \tag{8}$$

Finally, we set

$$b = \sum_{i=0}^{l}(I_i + R_i),\ s = \rho b + \sqrt{c_s \rho b \log f}, \tag{9}$$

where c_s is a positive constant. Recall that c_p is a function of c_d. To guarantee the nonnegativity of each phase in the policy, the constants c_d, c_s are chosen such that

$$\begin{aligned} 2c_p - \sqrt{6c_sc_p} \ge 1,\ c_d \ge 6c_p \ge 2c_p \ge c_s \ge 1280 \\ c_d + 2 \le \rho n^{0.5},\ n \ge 4,\ 6c_p \log f \le f \\ 40 \le \frac{c_d}{6c_p} \le c_d \le (1-\rho)^{-1} n^{-0.25}. \end{aligned} \tag{10}$$

Basically, as long as c_d and n are large enough, above constraints are easily satisfied.

5.5 Length Bound of Phases

Before we move forward to the policy analysis, we need to make sure the policy itself is well-defined. The following lemmas justify it in the sense that every phase has a positive number of slots. Throughout this section, we assume assumptions in Theorem 1 hold.

The first lemma gives bound on each phase length I_u and R_u in Recursive Clearing.

Lemma 1 *For each u such that $0 \le u \le l$, it holds that $I_u \ge \frac{1}{2}d \ge n$, and $R_u \ge n$.*

Proof For the bound on I_u, it suffices to prove it for I_l because the length I_u is decreasing. By definitions of l, I_u in (7) and (8), we have

$$\begin{aligned} I_l &= d - l\left(8\sqrt{19(c_d+2)}\right) n^{-0.25}(1-\rho)^{-1}\log f \\ &= d - \left\lfloor \frac{c_d}{16\sqrt{19(c_d+2)}} \right\rfloor \left(8\sqrt{19(c_d+2)}\right) n^{0.25}(1-\rho)^{-1}\log f \\ &\ge d - \left\lfloor \frac{c_d}{2} n^{0.25}(1-\rho)^{-1}\log f \right\rfloor \\ &\ge d - \frac{d}{2} = \frac{d}{2}. \end{aligned}$$

To see $\frac{d}{2} \geq n$, recall $d = c_d n^{0.25}(1-\rho)^{-1}\log f$. As $(1-\rho)^{-1} \geq n^{0.75}, c_d \geq 1280, \log f \geq 1$, we then have $d \geq 2n$.

To show $R_u \geq n$, we only need to show $\widetilde{R} \geq n$ as $R_u = \widetilde{R}$. By definition of $\widetilde{R}$, it holds

$$\widetilde{R} \geq 2n^{-0.5}(1-\rho)^{-2}\log f - n \geq 2n\log f - n = n,$$

where the first inequality is because $\widetilde{R}$ is the largest multiple of n that is less than $2n^{-0.5}(1-\rho)^{-2}\log f$, and the second inequality is because $(1-\rho)^{-1} \geq n^{0.75}$.

The next lemma gives a tight bound on the arrival period length b.

Lemma 2 *The period length b satisfies*

$$2c_p(1-\rho)^{-2}\log f \leq b \leq 6c_p(1-\rho)^{-2}\log f.$$

Proof By definition,

$$\begin{aligned} b &= \sum_{u=0}^{l} I_u + (l+1)R_0 \\ &\geq (l+1)\left(\frac{d}{2} + 2n^{-0.5}(1-\rho)^{-2}\log f - n\right) \\ &\geq 2c_p(1-\rho)^{-2}\log f, \end{aligned}$$

where we use Lemma 1 twice. Similarly, for the upper bound, we have

$$\begin{aligned} b &= \sum_{u=0}^{l} I_u + (l+1)R_0 \\ &\leq (l+1)\left(d + 2n^{-0.5}(1-\rho)^{-2}\log f\right) \\ &\leq 6c_p(1-\rho)^{-2}\log f, \end{aligned}$$

where the last inequality is because $d \leq n^{-0.5}(1-\rho)^{-2}\log f$ by the constraint $c_d \leq (1-\rho)^{-1}n^{-0.25}$ in (10).

Lemma 3 *The length of Backlog Clearing, $b-s$, is equal to $c_r(1-\rho)^{-1}\log f$ where $c_r = 2c_p - \sqrt{6c_s c_p}$ is larger or equal to 1.*

Proof We have

$$\begin{aligned} b - s &= b - \rho b - \sqrt{c_s \rho b \log f} \\ &\geq 2c_p(1-\rho)^{-1}\log f - \sqrt{6c_s c_p}(1-\rho^{-1})\log f \\ &= (2c_p - \sqrt{6c_s c_p})(1-\rho)^{-1}\log f. \end{aligned}$$

By constraints in (10), $c_r = 2c_p - \sqrt{6c_s c_p} \geq 1$.

The final lemma provides the length bound of Normal Clearing.

Lemma 4 *The length of Normal Clearing, $s-(b-d)$, is at least $1280n^{0.25}(1-\rho)^{-1}\log f$.*

Proof It holds that $s-(b-d) = c_d n^{0.25}(1-\rho)^{-1}\log f - c_r(1-\rho)^{-1}\log f$ which is at least $(c_d - c_r)n^{0.25}(1-\rho)^{-1}\log f$. We then complete the proof by noticing that $c_d - c_r = c_d - 2c_p + \sqrt{6c_s c_p} \geq \sqrt{6c_s c_p} \geq 1280$ because of (10).

6 Performance Analysis

In this section, we study the performance of the proposed policy. We first summarize a sketch of the whole proof which consists of multiple parts. Then we detail the proof of each part in separate sections. Proofs in this section have a similar structure as [17,25,15]. The key difference is that our analysis deals with two interchanging algorithms in Recursive Clearing while previous work only has one or no algorithm for this phase. Throughout this section, we assume conditions in Theorem 1 are satisfied. The switch size n is assumed to be large enough.

Proof Sketch

Our goal is to analyze the expected queue length under the policy described in Section 5. For each time slot τ, the key idea is to show the expected queue length $\mathbb{E}\left[\sum_{ij} Q_{ij}(\tau)\right]$ is of order $O(nd)$ where d is the length of the Idling phase.

Suppose the time slot τ lies in the k^{th} service period, i.e., it is in the range $[kb+d+1, (k+1)b+d]$ for some $k \in \mathbb{Z}^+$. For the first d slots $\{1, 2, \cdots, d\}$, the mean queue length is just bounded by ρnd since the mean arrival rate to each port is ρ. Since our analysis is on each arrival period, to simplify notations, define $A_{ij}^k(t)$ as the number of arrivals to queue Q_{ij}^k, and $S_{ij}^k(t)$ as the number of served packets in Q_{ij}^k during the first t time slots in arrival period k. That is, $A_{ij}^k(t) := A_{ij}(t+kb) - A_{ij}(kb)$, $S_{ij}^k(t) := S_{ij}(t+kb) - S_{ij}(kb)$. Let $\mathbf{A}^k(t) = (A_{ij}^k(t))_{n\times n}$.

By definition of $\mathbf{Q}^k$ and the backlog queue $\mathbf{B}$, it holds

$$\mathbb{E}\left[\mathbf{Q}(\tau)\right] = \mathbb{E}\left[\mathbf{Q}^k(\tau - kb)\right] + \mathbb{E}\left[B_k\right] + \mathbb{E}\left[\mathbf{A}^{k+1}((\tau-(k+1)b)^+)\right].$$

The last term is because new arrivals of the next period also contribute to the queue length. But as $\tau \leq (k+1)b+d$, the sum of elements in the last term is no larger than nd. Then by bounding $\mathbb{E}\left[\mathbf{Q}^k(\tau-kb)\right]$ and $\mathbb{E}\left[B_k\right]$ separately, we can bound the total queue length.

We first bound the first term. Suppose $t = \tau - kb$. It holds

$$\sum_{i=1}^{n}\sum_{j=1}^{n} Q_{ij}^k(t) = \sum_{i=1}^{n}\sum_{j=1}^{n} A_{ij}^k(t) - \sum_{i=1}^{n}\sum_{j=1}^{n} S_{ij}^k(t). \tag{11}$$

The expectation of the first term on the right hand side is equal to ρnt because packets arrive in a Bernoulli process. To bound the second term, we make use of the following key property: with high probability, we can schedule exactly n packets for each time slot during Recursive Clearing. The intuition behind this result is as follows. Fix an index u such that $0 \leq u \leq l$. Let $L = \sum_{p=0}^{u-1} T_p$ where $T_p = I_p + R_p$.

– For a Round-Robin phase R_u, the requirement to guarantee no wasted service is that $Q_{ij}^k(L+m)$ is positive for all $I_u + 1 \leq m \leq I_u + R_u, i, j \in [n]$. Since no packet arrives in I_u will be served (by the definition of the Idling phase and Packet-Collecting phases), it holds that with high probability, $Q_{ij}^k(L+m) \geq \left(\frac{\rho m}{n} - \sqrt{\frac{\rho m}{n}} \log n\right) - \frac{m-I_u}{n}$. The first term is because of the concentration property of Bernoulli random variables, and the second term is because we run Round-Robin in R_u. By carefully choosing phase length I_u, R_u, we can then guarantee $Q_{ij}^k(L+m)$ is positive throughout this Round-Robin phase.
– For a Packet-Collecting phase I_u with $1 \leq u \leq l$, the requirement is to guarantee $Q_{ij}^k(L)$ has an I_u-lower envelopes defined in 4.2. We use Theorem 3 to justify that such lower envelope exists with high probability. A key insight in the proof is to use the service regularity of Round-Robin algorithm in R_u. If we only look at packets that arrive in phase I_{u-1} and R_{u-1}, the mean number of arrivals is uniform among all queues. As R_{u-1} is a multiple of n, every queue in the switch will be scheduled for exactly $\frac{R_{u-1}}{n}$ times. Therefore, the mean queue lengths of all queues are almost uniform as well. This result indicates that the queue matrix $\mathbf{Q}^k(L)$ may have a similar structure as a random graph where the analysis of lower envelopes has been done in [25].

By above arguments, we can see $\mathbb{E}\left[\sum_{i,j} S_{ij}^k(t)\right]$ in (11) is equal to $n(\min(t,b)-d)$ with high probability. Therefore, the mean queue length in (11) can be bounded by $\rho nt - n(\min(t,b)-d) \leq 2nd$.

Finally, to show $\mathbb{E}[B_k]$ is small, we use an analysis similar to [17,25]. Recall the update equation (5) of B_k. We can view the backlog queue $\mathbf{B}$ as a discrete time G/G/1 queue. Then as long as we can show U_k is small, the famous Kingman bound [20] can be used to prove that $\mathbb{E}[B_k]$ is insignificant. To bound U_k, notice that if the policy can clear all packets in $Q^k(b)$ within Normal Clearing, then $U_k = 0$. By Theorem 2, we only require that the phase length $s - (b-d)$ is larger than or equal to the maximum number of packets of one port in $Q^k(b)$ to guarantee $U_k = 0$. Then we can once again use the equation (11), but now use it for one specific port. For any input port i, As argued before, with high probability, the total amount of service $\sum_j S_j^k(b)$ is equal to $b-d$. And the amount of arrivals to port i is less than $\rho b + \sqrt{\rho b \log n}$ with high probability by the concentration property of Bernoulli random variables. Then, by the definition of s in (9), we have $s-(b-d) \geq \rho b + \sqrt{\rho b \log n} - (b-d)$. As a result, $U_k = 0$ with high probability, which completes the whole proof.

6.1 Useful Facts

Before we proceed to the complete proof of our result, we first introduce several useful facts on which our analysis relies.

Kingman Bound for discrete-time G/G/1 queue

Consider a G/G/1 queue $\{Z(\tau), \tau \geq 0\}$ with an arrival process $\{X(\tau), \tau \geq 1\}$ and a service process $\{Y(\tau), \tau \geq 1\}$ where both $\{X(\tau), \tau \geq 1\}$ and $\{Y(\tau), \tau \geq 1\}$ consist of i.i.d. random variables, and the two processes are independent from each other. Suppose the queue evolves as

$$Z(\tau+1) = (Z(\tau) + X(\tau) - Y(\tau))^{+}.$$

Define $\lambda = \mathbb{E}\left[X(\tau)\right]$,$m_{2x} = \mathbb{E}\left[X^2(\tau)\right]$, $\mu = \mathbb{E}\left[Y(\tau)\right]$, $m_{2y} = \mathbb{E}\left[Y^2(\tau)\right]$. The following result is from [17, Theorem 4.2].

Theorem 4 *Suppose that $Z(0) = 0$ and that $\lambda < \mu$. Then*

$$\mathbb{E}\left[Z(\tau)\right] \leq \frac{m_{2x} + m_{2y} - 2\lambda\mu}{2(\mu - \lambda)}, \quad \textit{for all} \;\; \tau. \tag{12}$$

Concentration Inequality

The following result is adapted from [4, Theorem 2.4].

Theorem 5 *Let $X_1, \cdots, X_n$ be independent random variables with*

$$\mathbb{P}(X_i = 1) = p, \quad \mathbb{P}(X_i = 0) = 1 - p$$

for $i \in [n], p \in [0, 1]$. Let $X = \sum_{i=1}^{n} X_i$. Then for any $x > 0$, we have

$$\mathbb{P}(X \geq \mathbb{E}\left[X\right] + x) \leq \exp\left(-\frac{x^2}{2(\mathbb{E}\left[X\right] + x/3)}\right), \tag{13}$$

$$\mathbb{P}(X \leq \mathbb{E}\left[X\right] - x) \leq \exp\left(-\frac{x^2}{2\mathbb{E}\left[X\right]}\right). \tag{14}$$

6.2 Service Analysis

In this part, we show that with high probability, there is no wasted service during Recursive Clearing. We first present the analysis of Round-Robin phases, and then the analysis of Packet-Collecting phases. Throughput the analysis, we restrict the scope to the k^{th} arrival period and service period.

6.2.1 Round-Robin Phase

Suppose we are considering Round-Robin phase R_u where $0 \leq u \leq l$. Let $L = \sum_{j=0}^{u-1} T_j$ be the ending slot of R_{u-1} as shown in Figure 1. If $u = 0$, then $L = 0$. Recall that in a Round-Robin phase, we sequentially use a set of permutations defined in (6). The next lemma shows that under such policy, every queue Q_{ij}^k is always non-empty as long as there are enough arrivals.

Lemma 5 *Suppose $t \in \{L + I_u, \cdots, L + T_u - 1\}$. For any $i, j \in [n]$, if*

$$A_{ij}^k(t) - A_{ij}^k(L) > \frac{t - (L + I_u)}{n} + 1,$$

then $Q_{ij}^k(t+1) > 0$.

Proof It holds

$$\begin{aligned} Q_{ij}^k(t+1) &= A_{ij}^k(t) - S_{ij}^k(t) \\ &= A_{ij}^k(L) + A_{ij}^k(t) - A_{ij}^k(L) \\ &\quad - \left(S_{ij}^k(L + I_u) + S_{ij}^k(t) - S_{ij}^k(L + I_u)\right) \\ &\geq A_{ij}^k(t) - A_{ij}^k(L) - \left(S_{ij}^k(t) - S_{ij}^k(L + I_u)\right), \end{aligned}$$

where the last inequality is due to $A_{ij}^k(L) - S_{ij}^k(L + I_u) \geq 0$ since in I_u, we only schedule packets that arrive in the first L slots of the current period.

Then by the definition of Round-Robin policy in (6), every queue Q_{ij}^k will only be schedule once for every n slots. We thus have

$$S_{ij}^k(t) - S_{ij}^k(L + I_u) \leq \left\lceil \frac{t - (L + I_u)}{n} \right\rceil \leq \frac{t - (L + I_u)}{n} + 1.$$

Therefore, if $A_{ij}^k(t) - A_{ij}^k(L) > \frac{t-(L+I_u)}{n} + 1$, we have $Q_{ij}^k(t+1) > 0$.

As we can see, as long as the condition in Lemma 5 holds, the policy can schedule exactly n packets for every time slot in R_u. We now show that the condition holds with high probability. Define the event

$$\mathcal{W}_{ij}^k(t) = \left\{ A_{ij}^k(t) - A_{ij}^k(L) \leq \frac{t - (L + I_u)}{n} + 1 \right\}$$

for $t \in \{L + I_u, \cdots, L + T_u - 1\}$. Let $\mathcal{W}_u^k$ be the event such that the event $\mathcal{W}_{ij}^k(t)$ happens for some $i, j \in [n], t \in \{L + I_u, \cdots, L + T_u - 1\}$. Then

$$\mathcal{W}_u^k = \bigcup_{i=1}^{n} \bigcup_{j=1}^{n} \bigcup_{t=L+I_u}^{L+T_u-1} \mathcal{W}_{ij}^k(t).$$

Recall that $f = \max(n, (1-\rho)^{-1})$. We can show that $\mathcal{W}_u^k$ happens with tiny probability as in the next lemma.

Lemma 6 *For the event $\mathcal{W}_u^k$, it holds $\mathbb{P}\left(\mathcal{W}_u^k\right) \leq f^{-16}$.*

Proof We first bound the probability $\mathbb{P}\left(\mathcal{W}_{ij}^k(t)\right)$, and then use the union bound to get the desired result. Fix $i, j \in [n], t \in \{L + I_u, \cdots, L + T_u - 1\}$. Let $X = A_{ij}^k(t) - A_{ij}^k(L)$. Then X is the sum of $t - L$ i.i.d. Bernoulli random variables with $\mathbb{E}[X] = \frac{\rho(t-L)}{n}$. The event $\mathcal{W}_{ij}^k(t)$ is equivalent to

$$X \leq \frac{\rho(t-L)}{n} - \frac{(1-\rho)(t-L)}{n} - \frac{I_u}{n} + 1,$$

which can be rewritten as $\{X \leq \mathbb{E}[X] - x\}$ with $x = -(1-\rho)\frac{t-L}{n} + \frac{I_u}{n} - 1$. Notice that

$$\begin{aligned} x &\geq -(1-\rho)\frac{I_u + R_u}{n} + \frac{I_u}{n} - 1 \\ &= \frac{1}{n}\left(\rho I_u - (1-\rho)R_u - n\right) \\ &\geq \frac{1}{n}\left(\rho I_u - (1-\rho)T_u - n\right), \end{aligned}$$

where $T_u = I_u + R_u$. By definition, $T_u \leq b$, and thus $T_u \leq 6c_p(1-\rho)^{-2}\log f$ using Lemma 2. On the other hand, by Lemma 1, $I_u \geq \frac{1}{2}d = \frac{1}{2}c_d n^{0.25}(1-\rho)^{-1}\log f$. We have

$$x \geq \frac{1}{n}\left(\frac{1}{2}\rho c_d n^{0.25}(1-\rho)^{-1}\log f - 6c_p(1-\rho)^{-1}\log f - n\right).$$

By assumption, $(1-\rho)^{-1} \geq n^{0.75}$, and thus

$$x \geq \frac{1}{n^{0.75}}(1-\rho)^{-1}\log f\left(\frac{1}{2}\rho c_d - 6c_p n^{-0.25} - 1\right).$$

We claim that $\frac{1}{2}\rho c_d - 6c_p n^{-0.25} - 1 \geq \frac{1}{4}c_d$.

Proof (Proof of the claim) To prove the claim, seeing that

$$\begin{aligned} &\frac{1}{2}\rho c_d - 6c_p n^{-0.25} - 1 \geq \frac{1}{4}c_d \\ \Longleftarrow\ &\frac{1}{2}\rho\frac{c_d}{6c_p} - n^{-0.25} - \frac{1}{6c_p} \geq \frac{1}{4}\frac{c_d}{6c_p} \\ \Longleftarrow\ &\left(\frac{1}{2}\rho - \frac{1}{4}\right)\frac{c_d}{6c_p} \geq 2. \end{aligned} \tag{15}$$

Then by constraints in (10), we have $n \geq 4$, and thus $\rho \geq 1 - n^{-0.75} \geq 0.6$. Further with $\frac{c_d}{6c_p} \geq 40$, we can justify the last inequality in (15) which completes the proof.

As a result, we have

$$x \geq \frac{c_d}{4n^{0.75}}(1-\rho)^{-1}\log f.$$

On the other hand, $\mathbb{E}[X] = \frac{\rho(t-L)}{n} \leq \frac{T_u}{n}$. By the concentration bound in Theorem 5,

$$\begin{aligned}\mathbb{P}(X \leq \mathbb{E}[X] - x) &\leq \exp\left(-\frac{x^2}{2\mathbb{E}[X]}\right)\\ &\leq \exp\left(-\frac{c_d^2}{32}\log^2 f \cdot \frac{n^{-0.5}(1-\rho)^{-2}}{T_u}\right).\end{aligned}$$

Then notice that

$$\begin{aligned}T_u = I_u + R_u &\leq d + \widetilde{R}\\ &\leq c_d n^{0.25}(1-\rho)^{-1}\log f + 2n^{-0.5}(1-\rho)^{-2}\log f\\ &\leq (c_d+2)n^{-0.5}(1-\rho)^{-2}\log f.\end{aligned}$$

since $(1-\rho)^{-1} \geq n^{0.75}$. Therefore,

$$\begin{aligned}\mathbb{P}(X \leq \mathbb{E}[X] - x) &\leq \exp\left(-\frac{c_d^2}{32(c_d+2)}\log f\right)\\ &\leq f^{-21}.\end{aligned}$$

The last inequality is because $c_d \geq 1280$ and $\frac{c_d^2}{32(c_d+2)}$ is an increasing function. Then for every $i, j \in [n], t \in \{L + I_u, \cdots, L + T_u - 1\}$, it holds $\mathbb{P}\left(\mathcal{W}_{ij}^k(t)\right) \leq f^{-21}$. Notice that $b \leq 6c_p(1-\rho)^{-2}\log f$ by Lemma 2, and $6c_p \log f \leq f$ by constraints (10). Finally, by the union bound, we have

$$\mathbb{P}\left(\mathcal{W}_u^k\right) \leq n^2 b f^{-21} \leq f^{-16}.$$

6.2.2 Packet-Collecting Phase

We now proceed to the analysis of Packet-Collecting phases. Suppose we fix a Packet-Collecting phase I_u where $1 \leq u \leq l$. As before, let $L = \sum_{j=0}^{u-1} T_j$ which is the ending slot of the previous phase. To ensure that the policy can schedule exactly n packets for every slot in I_u, we require the existence of an I_u-lower envelope of the queue matrix $\mathbf{Q}^k(L+1)$. Define $\mathcal{P}_u^k$ as the event that $\mathbf{Q}^k(L+1)$ does not have an I_u-lower envelope. The following lemma shows that it is unlikely that $\mathcal{P}_u^k$ will happen.

Lemma 7 *For the event $\mathcal{P}_u^k$, it holds*

$$\mathbb{P}\left(\mathcal{P}_u^k\right) \leq f^{-16}.$$

This lemma is similar to [25, Theorem 5]. Indeed, [25, Theorem 5] shows that Condition (4) in Theorem 3 holds with high probability when $q_{ij} \sim \text{Binomal}(T_{u-1}, \frac{\rho}{n})$. In our algorithm, due to the effect of Round-Robin, we would subtract $\frac{R_{u-1}}{n}$ on each q_{ij}. Condition (4) would naturally hold since q_{ij} still roughly follows a binomial distribution, which completes the proof. Nevertheless, for completeness, we provide a formal proof as follows.

Proof The idea is to use Theorem 3. For any subset $\mathcal{R}, \mathcal{C}$ of $[n]$ and $t \in \{L - T_{u-1}, \cdots, L+1\}$, define

$$\begin{aligned} Q^k_{\mathcal{R},\mathcal{C}}(t) &= \sum_{i\in\mathcal{R}, j\in\mathcal{C}} Q^k_{ij}(t), A^k_{\mathcal{R},\mathcal{C}}(t) &= \sum_{i\in\mathcal{R}, j\in\mathcal{C}} A^k_{ij}(t), \\ S^k_{\mathcal{R},\mathcal{C}}(t) &= \sum_{i\in\mathcal{R}, j\in\mathcal{C}} S^k_{ij}(t). \end{aligned}$$

It suffices to show for any subset $\mathcal{R}, \mathcal{C} \subseteq [n]$, it holds

$$Q^k_{\mathcal{R},\mathcal{C}}(L+1) \geq I_u(|\mathcal{R}| + |\mathcal{C}| - n). \tag{16}$$

We can see

$$\begin{aligned} Q^k_{\mathcal{R},\mathcal{C}}(L+1) &= A^k_{\mathcal{R},\mathcal{C}}(L - T_{u-1}) + A^k_{\mathcal{R},\mathcal{C}}(L) - A^k_{\mathcal{R},\mathcal{C}}(L - T_{u-1}) \\ &\quad - \left(S^k_{\mathcal{R},\mathcal{C}}(L - R_{u-1}) + \S^k_{\mathcal{R},\mathcal{C}}(L) - S^k_{\mathcal{R},\mathcal{C}}(L - R_{u-1})\right) \\ &\geq A^k_{\mathcal{R},\mathcal{C}}(L) - A^k_{\mathcal{R},\mathcal{C}}(L - T_{u-1}) \\ &\quad - \left(S^k_{\mathcal{R},\mathcal{C}}(L) - S^k_{\mathcal{R},\mathcal{C}}(L - R_{u-1})\right). \end{aligned} \tag{17}$$

The last inequality holds because in I_{u-1}, the policy will only serve packets in $\mathbf{Q}^k(L - T_{u-1} + 1)$, and thus $A^k_{ij}(L - T_{u-1}) - S^k_{ij}(L - R_{u-1}) \geq 0$ for all $i, j \in [n]$.

Notice that R_{u-1} is a multiple of n by definition, and we run a Round-Robin policy (6) in $\{L - R_{u-1} + 1, \cdots, L\}$. We have

$$S^k_{\mathcal{R},\mathcal{C}}(L) - S^k_{\mathcal{R},\mathcal{C}}(L - R_{u-1}) = \frac{|\mathcal{R}||\mathcal{C}|R_{u-1}}{n}. \tag{18}$$

Define $X_{\mathcal{R},\mathcal{C}} = A^k_{\mathcal{R},\mathcal{C}}(L) - A^k_{\mathcal{R},\mathcal{C}}(L - T_{u-1})$. Then $X_{\mathcal{R},\mathcal{C}}$ is a Binomial random variable with parameters $T_{u-1}|\mathcal{R}||\mathcal{C}|$ and $\frac{\rho}{n}$.

Fix $|\mathcal{R}| = k, |\mathcal{C}| = m$. Let

$$\underline{\mathrm{X}}(k,m) = \min_{\mathcal{R},\mathcal{C}\subseteq[n]:\ |\mathcal{R}|=k, |\mathcal{C}|=m} X_{\mathcal{R},\mathcal{C}}.$$

Using (17) and (18), the condition (16) is satisfied if we have

$$\underline{\mathrm{X}}(k,m) - \frac{kmR_{u-1}}{n} \geq I_u\left(k + m - n\right) \tag{19}$$

for any $k, m \in [n]$.

Without loss of generality, assume $k \geq m$ and $k + m \geq n + 1$ since we can swap the role of input ports and output ports, and (19) trivially holds when $k + m \leq n$. To show that $\underline{\mathrm{X}}(k,m)$ is usually large, let us fix two subsets $\mathcal{R}, \mathcal{C} \subseteq [n]$ with $|\mathcal{R}| = k, |\mathcal{C}| = m$. Denote $p = \frac{\rho}{n}$. Recall that $X_{\mathcal{R},\mathcal{C}}$ is a binomial random variable with mean $kmT_{u-1}p$. By the concentration bound in Theorem 5, it holds

$$\begin{aligned}
&\mathbb{P}\left(X_{\mathcal{R},\mathcal{C}} \le kmT_{u-1}p - \sqrt{38(n-k+m)kmT_{u-1}p\log f}\right)\\
&\le \exp\left(-\frac{38(n-k+m)kmT_{u-1}p\log f}{2kmT_{u-1}p}\right)\\
&= f^{-19(n-k+m)}.
\end{aligned}$$

The number of such pair of $\mathcal{R},\mathcal{C}$ is equal to $\binom{n}{k}\binom{n}{m}$ which is bounded by n^{n-k+m}. Then by the union bound,

$$\begin{aligned}
&\mathbb{P}\left(\underline{\mathrm{X}}(k,m) \le kmT_{u-1}p - \sqrt{38(n-k+m)kmT_{u-1}p\log f}\right)\\
\le& \binom{n}{k}\binom{n}{m}\mathbb{P}\left(X_{\mathcal{R},\mathcal{C}} \le kmT_{u-1}p - \sqrt{38(n-k+m)kmT_{u-1}p\log f}\right)\\
\le& n^{n-k+m}f^{-19(n-k+m)}\\
\le& f^{-18(n-k+m)} \le f^{-18}.
\end{aligned}$$

To prove that (19) happens with high probability, it remains to show

$$kmT_{u-1}p - \sqrt{38(n-k+m)kmT_{u-1}p\log f} - \frac{kmR_{u-1}}{n} \ge I_u(k+m-n).$$

Dividing both side by m, it is equivalent to show

$$kT_{u-1}p - \sqrt{38\frac{n-k+m}{m}kT_{u-1}p\log f} - \frac{kR_{u-1}}{n} \ge I_u\frac{k+m-n}{m}.$$

Since $n < k+m$, we have $n-k+m \le 2m$, and $2-\frac{n}{k} = 1-\frac{n-k}{k} \ge 1-\frac{n-k}{m} = \frac{k+m-n}{m}$. It is thus sufficient to verify

$$kT_{u-1}p - 2\sqrt{19kT_{u-1}p\log f} - \frac{kR_{u-1}}{n} \ge I_u\left(2-\frac{n}{k}\right). \tag{20}$$

Let $x = \frac{k}{n}$, and recall that $p = \frac{\rho}{n}$. Then (20) can be rewritten as

$$\rho xT_{u-1} - 2\sqrt{19\rho xT_{u-1}\log f} - R_{u-1}x \ge I_u\left(2-\frac{1}{x}\right),$$

which can be further written as

$$x\left(I_{u-1} - (1-\rho)T_{u-1}\right) - \sqrt{76\rho T_{u-1}\log f}\sqrt{x} \ge I_u\left(2-\frac{1}{x}\right). \tag{21}$$

Notice that

$$T_{u-1} \le d + \widetilde{R} \le (c_d+2)n^{-0.5}(1-\rho)^{-2}\log f,$$

and thus

$$(1-\rho)T_{u-1} \le 4\sqrt{19\rho T_{u-1}\log f}$$

since $c_d + 2 \le 304\rho n^{0.5}$ by (10). By the definition of I_u in (8), it holds

$$\begin{aligned} I_u &= d - u\left(8\sqrt{19(c_d+2)}\right) n^{-0.25}(1-\rho)^{-1}\log f \\ &= I_{u-1} - 8\sqrt{19(c_d+2)}n^{-0.25}(1-\rho)^{-1}\log f \\ &\le I_{u-1} - 2\cdot 4\sqrt{19\rho T_{u-1}\log f} \\ &\le I_{u-1} - (1-\rho)T_{u-1} - 4\sqrt{19\rho T_{u-1}\log f}. \end{aligned} \tag{22}$$

Therefore, To show that (21) is true, we only need to show

$$\begin{aligned} &x\left(I_{u-1} - (1-\rho)T_{u-1}\right) - 2\sqrt{19\rho T_{u-1}\log f}\sqrt{x} \\ \ge &\left(I_{u-1} - (1-\rho)T_{u-1} - 4\sqrt{19\rho T_{u-1}\log f}\right)\left(2 - \frac{1}{x}\right). \end{aligned}$$

By manipulating terms, the above inequality is equivalent to

$$\begin{aligned} &\left(x + \frac{1}{x} - 2\right)\left(I_{u-1} - (1-\rho)T_{u-1}\right) \\ . \quad &\ge 2\left(\sqrt{x} + \frac{2}{x} - 4\right)\sqrt{19\rho T_{u-1}\log f}. \end{aligned} \tag{23}$$

As $k + m > n$ and $k \ge m$, we have $x = \frac{k}{n} \ge \frac{1}{2}$. As a result, $x + \frac{1}{x} - 2 \ge \frac{1}{2}(\sqrt{x} + \frac{2}{x} - 4)$. Combining inequality (22) with the fact that $I_u \ge \frac{d}{2}$ shown in Lemma 1, it holds

$$I_{u-1} - (1-\rho)T_{u-1} \ge 4\sqrt{19\rho T_{u-1}\log f}.$$

Therefore, we establish (23), and thus (19). We now have shown

$$\mathbb{P}\left(\underline{\mathrm{X}}(k,m) \le I_u(k+m-n)\right) \le f^{-18}.$$

Note that we have assumed $k \ge m$, but swapping k, m does not affect the result.

Finally, to complete the whole proof, we use the union bound. Note that

$$\mathcal{P}_u^k = \bigcup_{k=1}^{n}\bigcup_{m=1}^{n}\mathbb{P}\left(\underline{\mathrm{X}}(k,m) \le I_u(k+m-n)\right).$$

As a result,

$$\mathbb{P}\left(\mathcal{P}_u^k\right) \le n^2 f^{-18} \le f^{-16}.$$

6.3 Backlog Analysis

In this section, we bound the expected number of backlogs, $\mathbb{E}[B_k]$. The analysis is similar to [17,25]. We include it here for completeness.

We first show that for any arrival period k where $k \in \mathbb{Z}^+$, there is a high probability that all packets in arrival period k will be cleared in Normal Clearing. Fix $k \in \mathbb{Z}^+$. For any $i, j \in [n]$, define

$$H_i^k = \sum_{j'=1}^{n} A_{ij'}^k(b), \quad C_j^k = \sum_{i'=1}^{n} A_{i'j}^k(b),$$

which are the total number of arrivals to input port i and the total number of arrivals to output port j during the arrival period k, respectively.

Define the event

$$\mathcal{E}_k = \left\{\exists_{i\in[n]},\ H_i^k > s\right\} \cup \left\{\exists_{j\in[n]},\ C_j^k > s\right\}$$

as the event when some ports may receive excessive packets. The following lemma shows that such event is rare.

Lemma 8 *For all $k \in \mathbb{Z}^+$, we have $\mathbb{P}(\mathcal{E}_k) \le \frac{1}{2} f^{-13}$.*

Proof First consider the event $\{H_1^k > s\}$. As H_1^k is a binomial random variable with parameter nb and $\frac{\rho}{n}$, the concentration bound in Theorem 5 implies that

$$\begin{aligned}
\mathbb{P}\left(H_1^k > s\right) &= \mathbb{P}\left(H_1^k > \rho b + \sqrt{c_s b \log f}\right)\\
&= \mathbb{P}\left(H_1^k > \mathbb{E}\left[H_1^k\right] + \sqrt{c_s b \log f}\right)\\
&\le \exp\left(-\frac{c_s b \log f}{2(\rho b + \sqrt{c_s b \log f}/3)}\right)\\
&\le \exp\left(-\frac{c_s b \log f}{2s}\right)\\
&\le \exp\left(-\frac{c_s b \log f}{2b}\right)\\
&\le f^{-c_s/2}.
\end{aligned}$$

As c_s is assumed to be at least 30 in (10), we have $\mathbb{P}\left(H_1^k > s\right) \le f^{-15}$. Since arrival rates are uniform among each pair of input ports and output ports, it holds that for every $i, j \in [n]$,

$$\mathbb{P}\left(H_i^k > s\right) \le f^{-15}, \quad \mathbb{P}\left(C_j^k > s\right) \le f^{-15}.$$

Then by the union bound, we have

$$\mathbb{P}\left(\mathcal{E}^k\right) \le 2nf^{-15} \le \frac{1}{2} f^{-13}$$

because we assume $n \ge 4$ in (10).

Now we combine above arguments with previous analysis of Recursive Clearing. Define

$$\mathcal{W}^k = \bigcup_{u=0}^{l} \mathcal{W}_u^k, \quad \mathcal{P}^k = \bigcup_{u=1}^{l} \mathcal{P}_u^k.$$

We have the following lemma claiming that $U_k = 0$ with high probability.

Lemma 9 *For a fixed k, the following results hold.*

1. *The number of remaining packets U_k is zero if none of $\mathcal{W}^k$, $\mathcal{P}^k$, $\mathcal{E}^k$ occurs.*
2. *The probability $\mathbb{P}\left(\mathcal{W}^k \cup \mathcal{P}^k \cup \mathcal{E}^k\right)$ is bounded by f^{-13}, and thus $\mathbb{P}\{U_k > 0\} \leq f^{-13}$.*
3. *On any sample path, $U_k \leq n^2 b$.*

Proof For a fixed k and $i, j \in [n]$, define

$$\widetilde{H}_i = \sum_{j'=1}^{n} Q_{ij'}^k(b+1), \quad \widetilde{C}_j = \sum_{i'=1}^{n} Q_{i'j}^k(b+1).$$

By the optimal clearing policy in Theorem 2, if $\gamma = \max\left(\max_i \widetilde{H}_i, \max_j \widetilde{C}_j\right)$ is no larger than the length of Normal Clearing, $s - b + d$, then there exists a scheduling that can clear all packets in $\mathbf{Q}^k(b+1)$ within Normal Clearing.

To prove the first result, assume that none of $\mathcal{W}^k, \mathcal{C}^k, \mathcal{E}^k$ occurs. Notice that if neither $\mathcal{W}^k$ nor $\mathcal{P}^k$ occurs, then for every time slot τ in $\{kb + d + 1, \cdots, (k+1)b\}$, the policy will schedule exactly n packets. To justify this claim, suppose τ is in a Round-Robin phase R_u. When $\mathcal{W}^k$ does not happen, by Lemma 5, all queues $Q_{ij}^k(\tau - kb)$ have at least one packet. Any schedule that is perfect matching can serve exactly one packet from each input port, and to each output port. On the other hand, suppose τ is in a Packet-Collecting phase I_u. Since $\mathcal{P}^k$ does not occur, schedules in I_u form an I_u-lower envelope of the queue matrix at the beginning of this phase. By the definition of an I_u-lower envelope, the policy can schedule exactly n packets at time slot τ. Therefore, for any $i, j \in [n]$, it holds

$$\sum_{j'=1}^{n} S_{ij'}^k(b) = b - d, \quad \sum_{i'=1}^{n} S_{i'j}^k(b) = b - d.$$

Moreover, as $\mathcal{E}^k$ does not occur, we know $H_i^k \leq s, C_j^k \leq s$ for any $i, j \in [n]$. Therefore, for any $i \in [n]$,

$$\widetilde{H}_i = \sum_{j'=1}^{n} A_{ij'}^k(b) - \sum_{j'=1}^{n} S_{ij'}^k(b) = H_i^k - (b - d) \leq s - b + d.$$

Similarly, we have $\widetilde{C}_j \leq s - b + d$ for any $j \in [n]$. As a result, the maximum γ among $\widetilde{H}_i, \widetilde{C}_j$ is upper bounded by $s - b + d$. Using Theorem 2, we know there exists a sequence of γ matchings that can clear the queue matrix $\mathbf{Q}^k(b+1)$.

As $s - b + d \geq \gamma$, we can schedule these matchings during Normal Clearing, and thus $U_k = 0$.

We can now prove the second result. By Lemma 6, we have $\mathbb{P}\{\mathcal{W}_u^k\} \leq f^{-16}$ for any $0 \leq u \leq \ell$. Then by the union bound, it holds

$$\mathbb{P}\{\mathcal{W}^k\} \leq \ell f^{-16} = c_p n^{0.5} f^{-16} \leq f^{-15}$$

because $c_p \leq c_d \leq \rho n^{0.5}$ in (10). Similarly, we can bound $\mathbb{P}\{\mathcal{P}^k\}$ by

$$\mathbb{P}\{\mathcal{P}^k\} \leq \ell f^{-16} \leq f^{-15}$$

using Lemma 7. Together with Lemma 8, it holds

$$\mathbb{P}\left(U_k > 0\right) \leq \mathbb{P}\left(\mathcal{W}^k \cup \mathcal{P}^k \cup \mathcal{E}^k\right) \leq 2f^{-15} + \frac{1}{2}f^{-13} \leq f^{-13}.$$

Finally, to prove the third result, we have

$$U_k \leq \sum_{i=1}^{n}\sum_{j=1}^{n} A_{ij}^k(b) \leq n^2 b$$

because every queue has at most one arrival per time slot in a Bernoulli arrival process.

Based on the above result, we can bound the expected number of backlogs in the backlog queue $\mathbf{B}$.

Lemma 10 *It holds* $\mathbb{E}\left[B_k\right] \leq 1$ *for all* k.

Proof Recall that the queue length B_k updates as

$$B_{k+1} = \left(B_k + U_k - (b - s)\right)^+$$

with $B_0 = 0$. As $b - s \geq 1$ by Lemma 3, B_k is stochastically dominated by $\widetilde{B}_k$ which evolves as

$$\widetilde{B}_{k+1} = \left(B_k + U_k - 1\right)^+$$

where $\widetilde{B}_0 = 0$. The new process can be viewed as a discrete-time G/G/1 queue, and thus Kingman bound Theorem 4 applies. Using the same notation as in Theorem 4, we have $\mu = 1, m_{2y} = 1$. By Lemma 9, it holds

$$\lambda = \mathbb{E}\left[U_k\right] \leq n^2 b \mathbb{P}\left(U_k > 0\right) \leq n^2 b f^{-13} \leq f^{-8}$$

because $b \leq 6c_p(1-\rho)^{-2}\log f$ by Lemma 2, and $6c_p \log f \leq f$ by (10). Similarly,

$$m_{2x} = \mathbb{E}\left[U_k^2\right] \leq n^4 b^2 f^{-13} \leq f^{-3}.$$

As a result,

$$\mathbb{E}\left[B_k\right] \leq \mathbb{E}\left[\widetilde{B}_k\right] \leq \frac{m_{2x} + m_{2y} - 2\lambda\mu}{2(\mu - \lambda)} \leq \frac{f^{-3} + 1}{2(1 - f^{-8})} \leq 1.$$

6.4 Queue Length Analysis

This section presents the formal proof for the bound of the average total queue length in Theorem 1.

Lemma 11 *It holds that for every time slot τ,*

$$\mathbb{E}\left[\sum_{i=1}^{n}\sum_{j=1}^{n} Q_{ij}(\tau)\right] \leq 3nd.$$

Proof Fix a time slot τ, and we bound the expected queue lengths in $\mathbf{Q}(\tau)$. First, if $1 \leq \tau \leq d$, then

$$\mathbb{E}\left[\sum_{i=1}^{n}\sum_{j=1}^{n} Q_{ij}(\tau)\right] \leq \sum_{i=1}^{n}\sum_{j=1}^{n} \mathbb{E}\left[A_{ij}(\tau-1)\right] \leq \rho n\tau \leq nd.$$

We can thus assume $\tau \in [kb+d+1, (k+1)b+d]$ for some $k \in \mathbb{Z}^+$, i.e., τ is in the k^{th} service period. We consider different cases.

First, if $kb+d+1 \leq \tau \leq (k+1)b$, then τ is in Recursive Clearing. As a result,

$$\sum_{i=1}^{n}\sum_{j=1}^{n} Q_{ij}(\tau) = B_k + \sum_{i=1}^{n}\sum_{j=1}^{n} Q_{ij}^{k}(\tau - kb).$$

Let $t = \tau - kb$. We have

$$\begin{aligned}\mathbb{E}\left[\sum_{i=1}^{n}\sum_{j=1}^{n} Q_{ij}^{k}(t)\right] &= \mathbb{E}\left[\sum_{i=1}^{n}\sum_{j=1}^{n} A_{ij}^{k}(t-1) - \sum_{i=1}^{n}\sum_{j=1}^{n} S_{ij}^{k}(t-1)\right]\\ &= \rho n(t-1) - \mathbb{E}\left[\sum_{i=1}^{n}\sum_{j=1}^{n} S_{ij}^{k}(t-1)\right].\end{aligned}$$

As in the proof of Lemma 9, if neither $\mathcal{W}^k$ nor $\mathcal{P}^k$ happens, we have

$$\sum_{i=1}^{n}\sum_{j=1}^{n} S_{ij}^{k}(t-1) = (t-1-d)n.$$

Since $\mathbb{P}\left(\mathcal{W}^k \cup \mathcal{P}^k\right) \leq f^{-13}$ by Lemma 9, we have

$$\begin{aligned}\mathbb{E}\left[\sum_{i=1}^{n}\sum_{j=1}^{n} Q_{ij}^{k}(t)\right] &\leq \rho(t-1)n - (t-d-1)n\left(1-f^{-13}\right)\\ &\leq \rho(t-1)n - \rho(t-d-1)n\\ &= \rho nd.\end{aligned}$$

Together with Lemma 3, we have

$$\mathbb{E}\left[\sum_{i=1}^{n}\sum_{j=1}^{n} Q_{ij}(\tau)\right] \leq \mathbb{E}\left[B_k\right] + \rho nd \leq 1 + \rho nd \leq 2nd.$$

Consider the second case where $(k+1)b < \tau \leq (k+1)b + d$. The time slot τ is thus in Normal Clearing or Backlog Clearing. We can see

$$\begin{aligned}\mathbb{E}\left[\sum_{i=1}^{n}\sum_{j=1}^{n} Q_{ij}(\tau)\right] &\leq \mathbb{E}\left[\sum_{i=1}^{n}\sum_{j=1}^{n} Q_{ij}((k+1)b)\right] \\ &+ \mathbb{E}\left[\sum_{i=1}^{n}\sum_{j=1}^{n} \left(A_{ij}(\tau-1) - A_{ij}\left((k+1)b-1\right)\right)\right] \\ &\leq 2nd + nd = 3nd.\end{aligned}$$

Summarizing above discussions completes the proof.

6.5 Generalization to Poisson Arrivals

We remark that our queue length bound can be naturally generalized to other arrival processes, such as Poisson arrivals. In particular, since our proof does not make use of the boundedness of Bernoulli random variables, it is sufficient to generalize our result if we could have a similar concentration bound as Theorem 5. Indeed, recall that the sum of n i.i.d. Poisson random variables of rate λ is again a Poisson random variables but with rate $n\lambda$. Therefore, the concentration of the sum is a concentration of a Poisson random variable. Indeed, we have the following concentration bound for a Poisson random variable from [3], whose form is very much similar to Theorem 5.

Theorem 6 *Let $X \sim \text{Poisson}(\lambda)$ for $\lambda > 0$. Then for any $x > 0$, it holds*

$$\mathbb{P}\left(|X-\lambda| > x\right) \leq 2e^{\frac{-x^2}{2(\lambda+x)}}. \tag{24}$$

Our proof above would then hold by replacing the use of Theorem 5 by Theorem 6.

7 Complexity Analysis

This section analyzes the time complexity of the proposed policy with a discussion on the delay-complexity trade-offs. Results in this section also conclude the proof of Theorem 1.

7.1 Time Complexity

To calculate the average time complexity per time slot, our approach is to sum up all computation requirement during one service period, and then divide the sum by the period length b. The following lemma presents the amortized time complexity of our policy described in Section 5.

Lemma 12 *With the same setting in Theorem 1, the amortized time complexity of the policy in Section 5 is* $O(n + (1-\rho)^2 n^{3.5}/\log(f))$.

Proof As the policy is fixed in each service period, and each service period is the same, the average computation in one service period is exactly the amortized time complexity of the whole policy. Fix a service period k. We study the complexity in each phase separately.

1. For a Round-Robin phase R_u, the scheduling policy at one time slot τ can be calculated in $O(n)$ by the definition of a Round-Robin policy (6).
2. For a Packet-Collecting phase I_u, we need to calculate an I_u-lower envelope at the beginning of this phase. Recall the algorithm introduced in Section 4.2. The total complexity to verify the existence of an I_u-lower envelope and to find out one solution is $O(n^3 + nI_u \log n)$. Since $I_u \in [\frac{d}{2}, d]$ by Lemma 1, the time complexity for one Packet-Collecting phase is $O(n^3 + nd \log n)$. Note that if there is no such lower envelope, the policy does nothing by definition. It will not change the time complexity because such events are rare by Lemma 7.
3. For Normal Clearing, we consider the expected time complexity to find an optimal clearing policy. Through the discussion in 4.1, the time complexity is $O(m \log n)$ where m is the sum of all elements in $\mathbf{Q}^k(b+1)$. In the policy, we first check whether the maximum number of packets at each port is below the phase length. It takes $O(n^2)$ time to check the maximum number of packets. If the maximum exceeds the phase length, we directly skip finding the optimal clearing policy. Otherwise, when that number is below the phase length, the number of packets in $\mathbf{Q}^k(b+1)$ is bounded by $n(s-b+d)$. The time complexity in this case is thus $O(n^2+n(s-b+d)\log n)$, which is indeed $O(n(s-b+d)\log n)$ by Lemma 4.
   The final algorithm in Normal Clearing is to move packets in U_k into the backlog queue $\mathbf{B}$. However, as we could see, every incoming packets to the switch will be put into the backlog queue for at most once. The amortized complexity to move packets to the backlog queue $\mathbf{B}$ is $O(n)$ because by the law of large numbers, only ρn packets will join the switch in time average. The total computation to move packets into the backlog queue is thus $O(n(s-b+d))$ in Normal Clearing. To sum up, the total time complexity in this phase is bounded by $O(n(s-b+d)\log n)$.
4. Finally, for Backlog Clearing, we only schedule at most one packet from the backlog queue. Therefore, the total computation is bounded by $O(b-s)$.

As a result, the total computation in a service period k is given by

$$\sum_{u=1}^{\ell} O(nR_u) + \sum_{u=1}^{\ell} O(n^3 + nd \log n) + O(n(s-b+d)\log n) + O(b-s),$$

which is equal to

$$O(nb + n^{7/2} + n^{7/4}(1-\rho)^{-1} \log n \log f) \tag{25}$$

because $\ell = O(n^{1/2}), d = O(n^{1/4}(1-\rho)^{-1} \log f), s - b + d = O(d)$ by their definition in (7) and Lemma 4.

The amortized time complexity per slot in one service period is thus equal to (25) divided by b, which is

$$\begin{aligned}
&O\left(\frac{n(1-\rho)^{-2}\log f + n^{7/2} + n^{7/4}(1-\rho)^{-1}\log n \log f}{(1-\rho)^{-2}\log f}\right) \\
&= O(n) + O\left(\frac{n^{7/2}}{(1-\rho)^{-2}\log f}\right) + O\left(\frac{n^{7/4}\log n}{(1-\rho)^{-1}}\right).
\end{aligned}$$

We now bound the last two terms. If $\frac{n^{7/2}}{(1-\rho)^{-2}\log f} < \frac{n^{7/4}\log n}{(1-\rho)^{-1}}$, it immediately implies $n^{7/4}/\log f < (1-\rho)^{-1}\log n$. But in this case,

$$\frac{n^{7/4}\log n}{(1-\rho)^{-1}} < \frac{(1-\rho)^{-1}\log n \log f}{(1-\rho)^{-1}} = \log n \log f.$$

Then if $\log n \log f = \omega(n)$, we have $\log f = \omega(n/\log n)$. Note that $f = \max(n, (1-\rho)^{-1})$. It thus hold $(1-\rho)^{-1} = \exp(\omega(n/\log n))$. As a result, $\frac{n^{7/4}\log n}{(1-\rho)^{-1}} = o(1)$. We then have

$$\begin{aligned}
&O(n) + O\left(\frac{n^{7/2}}{(1-\rho)^{-2}\log f}\right) + O\left(\frac{n^{7/4}\log n}{(1-\rho)^{-1}}\right) \\
=&O(n) + O\left(\frac{n^{7/2}}{(1-\rho)^{-2}\log f}\right),
\end{aligned}$$

which completes the proof.

Lemma 12 shows that, if n is fixed, the complexity of the policy is decreasing as the traffic becomes heavier. The main reason is that, the policy will spend more time following Round-Robin policies, and less time finding lower envelopes when we have a larger ρ. Since Round-Robin policies takes $O(n)$ computation time instead of $O(n^3)$ time needed to find lower envelopes, the amortized complexity is reduced.

Using Lemmas 11 and 12, we can finish the proof of our main theorem, Theorem 1.

Proof (Proof of Theorem 1) The proposed policy in Section 5 has an average total queue length $O(n^{5/4}(1-\rho)^{-1}\log f)$ by Lemma 11, and its amortized time complexity is $O(n + n^{7/2}(1-\rho)^2/\log f)$ by Lemma 12. Therefore, the policy satisfies requirements in Theorem 1, which concludes the proof.

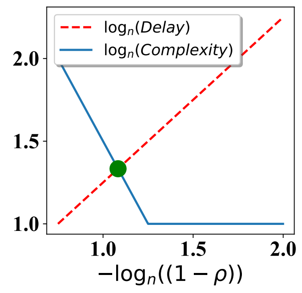

Fig. 2 Trade-offs between Delay and Complexity

7.2 Delay-Complexity Trade-offs

From Lemma 12 and Little's Law, the average delay is of order $O(n^{1/4}(1-\rho)^{-1}\log f)$, while the amortized time complexity is $O(n+n^{7/2}(1-\rho)^2/\log f)$. Suppose n is fixed, and it is large enough. Assume $\rho = 1-n^{-\alpha}$, and $\alpha > 0.75$. Then by changing α, we can plot the curve of delay and time complexity (taking logarithm with base n) as in Figure 2. The dot in Figure 2 refers to $\alpha^* = 1+\frac{1}{12}$. When $\alpha < \alpha^*$, we can see that the average delay is strictly below the amortized time complexity. Therefore, to improve the overall performance, we can reduce the complexity of the algorithm by trading off certain extent of delay performance. One solution is to adjust the traffic intensity ρ by introducing a stream of pseudo arrivals. Scheduling packets in this new environment can then automatically decrease the time complexity. Certainly, when $\alpha > \alpha^*$, the average delay dominates the time complexity. In this case, increasing the traffic intensity in our system may not help a lot.

We remark that other batching policies may have similar delay-complexity trade-offs. For example, the recent work [25] has an average delay $O((1-\rho)^{-4/3}\log f)$ for all $\rho < 1$. Although this work did not provide the time complexity of the algorithm, it can be bounded by $O(n\log n + n^3(1-\rho)^{4/3}/\log f)$ through the same technique in Section 7.1. We can see increasing the traffic intensity can help reduce its time complexity when the traffic is light.

8 Conclusion

In this paper, we present a new batch-scheduling algorithm for an input-queued switch with uniform Bernoulli arrivals. The key requirement of such a policy is to ensure full service at every time slot in Recursive Clearing shown in Figure 1; thus we need to wait for a sufficiently long period of time in the Idling phase to help the number of packet arrivals concentrate around their means. To help reduce the length of the Idling phase, this work successfully combines two kinds of phases, Round-Robin phases $\{R_u, u \geq 0\}$ and Packet-Collecting phases $\{I_u, u \geq 1\}$ where a phase I_u can be viewed as an Idling phase of R_u, and concentration of the number of arrivals around their means in I_u and R_u

and the regularity of service in R_u helps with full service in I_{u+1}. With a more effective scheduling algorithm, our policy thus enjoys a better average queue length $O(n^{5/4}(1-\rho)^{-1}\log f)$ than previous best known results in the regime $n^{3/4} \leq (1-\rho)^{-1} < n^{7/4}$. In particular, when $\rho = 1 - 1/n$, our result is $O(n^{2+1/4}\log n)$, while previous result is $O(n^{2+1/3}\log n)$ [25].

Acknowledgements The work of Wentao Weng was conducted during a visit to the Coordinated Science Lab, UIUC during 2020 when he was an undergraduate at Tsinghua University. Research is also supported in part by NSF Grants CPS ECCS 17-39189, CCF 19-34986, ECCS 16-09370, ONR Grant Navy N00014-19-1-2566, ARO Grant W911NF-19-1-0379.

References

1. Aggarwal, G., Motwani, R., Shah, D., Zhu, A.: Switch scheduling via randomized edge coloring. In: Proc. Ann. ACM Symp. Theory of Computing (STOC), pp. 502–512. IEEE (2003)
2. Andrews, M., Jung, K., Stolyar, A.: Stability of the max-weight routing and scheduling protocol in dynamic networks and at critical loads. In: Proc. Ann. ACM Symp. Theory of Computing (STOC), pp. 145–154 (2007)
3. Canonne, C.: A short note on poisson tail bounds. `https://github.com/ccanonne/probabilitydistributiontoolbox/blob/master/poissonconcentration.pdf` (2019)
4. Chung, F., Chung, F.R., Graham, F.C., Lu, L., Chung, K.F., et al.: Complex graphs and networks. 107. American Mathematical Soc. (2006)
5. Cole, R., Hopcroft, J.: On edge coloring bipartite graphs. Siam J. Comput. **11**(3), 540–546 (1982)
6. Dai, J., Prabhakar, B.: The throughput of data switches with and without speedup. In: Proc. IEEE Int. Conf. Computer Communications (INFOCOM), vol. 2, pp. 556–564. IEEE (2000)
7. Eryilmaz, A., Srikant, R.: Asymptotically tight steady-state queue length bounds implied by drift conditions. Queueing Syst. **72**(3-4), 311–359 (2012)
8. Goldberg, A.V., Tarjan, R.E.: A new approach to the maximum-flow problem. J. ACM **35**(4), 921–940 (1988)
9. Goldberg, A.V., Tarjan, R.E.: Efficient maximum flow algorithms. ACM Commun. **57**(8), 82–89 (2014)
10. Gupta, G.R., Sanghavi, S., Shroff, N.B.: Node weighted scheduling. In: Proc. ACM SIGMETRICS/PERFORMANCE Jt. Int. Conf. Measurement and Modeling of Computer Systems, pp. 97–108 (2009)
11. Jhunjhunwala, P., Maguluri, S.T.: Low-complexity switch scheduling algorithms: Delay optimality in heavy traffic. arXiv preprint arXiv:2004.12271 (2020)
12. Maguluri, S.T., Srikant, R.: Heavy traffic queue length behavior in a switch under the maxweight algorithm. Stoch. Syst. **6**(1), 211–250 (2016)
13. McKeown, N., Mekkittikul, A., Anantharam, V., Walrand, J.: Achieving 100% throughput in an input-queued switch. IEEE Trans. Commun. **47**(8), 1260–1267 (1999)
14. Neely, M.J.: Delay analysis for maximal scheduling in wireless networks with bursty traffic. In: Proc. IEEE Int. Conf. Computer Communications (INFOCOM), pp. 6–10. IEEE (2008)
15. Neely, M.J., Modiano, E., Cheng, Y.S.: Logarithmic delay for $n \times n$ packet switches under the crossbar constraint. IEEE/ACM Trans. Netw. **15**(3), 657–668 (2007)
16. Shah, D.: Maximal matching scheduling is good enough. In: Proc. IEEE Glob. Telecomm. Conf. (GLOBECOM), vol. 6, pp. 3009–3013. IEEE (2003)
17. Shah, D., Tsitsiklis, J.N., Zhong, Y.: On queue-size scaling for input-queued switches. Stoch. Syst. **6**(1), 1–25 (2016)
18. Shah, D., Walton, N.S., Zhong, Y., et al.: Optimal queue-size scaling in switched networks. Ann. Appl. Probab. **24**(6), 2207–2245 (2014)

19. Shah, D., Wischik, D., et al.: Switched networks with maximum weight policies: Fluid approximation and multiplicative state space collapse. Ann. Appl. Probab. **22**(1), 70–127 (2012)
20. Srikant, R., Ying, L.: Communication networks: an optimization, control, and stochastic networks perspective. Cambridge University Press (2013)
21. Stolyar, A.L., et al.: Maxweight scheduling in a generalized switch: State space collapse and workload minimization in heavy traffic. Ann. Appl. Probab. **14**(1), 1–53 (2004)
22. Tassiulas, L., Ephremides, A.: Stability properties of constrained queueing systems and scheduling policies for maximum throughput in multihop radio networks. In: Proc. IEEE Conf. Decision and Control (CDC), pp. 2130–2132. IEEE (1990)
23. Wang, L., Ye, T., Lee, T.T., Hu, W.: A parallel complex coloring algorithm for scheduling of input-queued switches. IEEE Trans. Parallel Distrib. Syst. **29**(7), 1456–1468 (2018)
24. Weller, T., Hajek, B.: Scheduling nonuniform traffic in a packet-switching system with small propagation delay. IEEE/ACM Trans. Netw. **5**(6), 813–823 (1997)
25. Xu, J., Zhong, Y.: Improved queue-size scaling for input-queued switches via graph factorization. Adv. Appl. Probab. **52**(3), 798–824 (2020)