Reduced-State, Optimal Scheduling for Decentralized Medium Access Control of a Class of Wireless Networks

Avinash Mohan, *Member, IEEE,* Aditya Gopalan, *Member, IEEE,* and Anurag Kumar, *Fellow, IEEE.*

***Abstract*—Motivated by medium access control for resource-challenged wireless Internet of Things (IoT) networks, we consider the problem of queue scheduling with reduced queue state information. In particular, we consider a time-slotted scheduling model with N wireless links, such that links i and $i+1$, $1 \leq i \leq N-1$ cannot transmit together. Our aim in this paper is to study throughput-optimal, and even delay optimal, scheduling policies that require only the empty-nonempty state of the packet queues associated with these links (Queue Nonemptiness Based, or QNB, policies). We focus on Maximum Size Matching (MSM) policies, and provide an analysis of all the QNB-MSM policies for $N = 3$, thereby comparing their performance, and revisiting a delay optimal scheduling result stated in [1]. Our study shows that, while scheduling a maximum size matching would seem intuitive, there are important performance differences between different QNB-MSM policies. Further, it is not necessary for a QNB policy to be MSM for it to be throughput optimal. We develop a new *Policy Splicing* technique to combine scheduling policies for small networks to construct throughput-optimal policies for larger networks, some of which also aim for low delay. For $N = 3$ there exists a QNB-MSM policy that is sum-queue optimal over the entire stability region. We show, however, that for $N \geq 4$, there is no QNB scheduling policy that is sum-queue length optimal over all arrival rate vectors in the capacity region.**

Our throughput-optimality results rely on two new arguments: a Lyapunov drift lemma specially adapted to policies that are queue length-agnostic, and a priority queueing analysis for showing strong stability. We then extend our results to a more general class of interference constraints that we call cluster-of-cliques (CoC) conflict graphs. We consider two types of CoC networks, namely, Linear Arrays of Cliques (LAoC) and Star-of-Cliques (SoC) networks. We develop QNB policies for these classes of networks, study their stability and delay properties, and propose and analyze techniques to reduce the amount of state information to be disseminated across the network for scheduling.

***Index Terms*—Wireless sensor networks, Medium Access Control (MAC) protocols, resource challenged networks, low delay wireless scheduling, Internet of Things.**

AM (*corresponding author*) is with the Technion Israel Institute Technology. AG and AK are with the Indian Institute of Science, Bangalore. This work was carried out when the first author was at the Indian Institute of Science. e-mail: avinashmohan@campus.technion.ac.il,{aditya, anurag}@iisc.ac.in.

This work was presented, in part, at the 13th IEEE International Conference on Computer Networks (IEEE Infocom, 2018). This research was supported by the Ministry of Human Resource Development Govt. of India, via a graduate fellowship for the first author, by Microsoft Research India, by a travel grant for the first author, by the SERB grant EMR/2016/002503 and the IUSSTF WAQM 2017 program for the second author, and the Department of Science and Technology, via a J.C. Bose Fellowship awarded to the third author.

Please note that all appendices are provided in the Supplementary Material.

I. Introduction

The Internet of Things (IoT) paradigm is expected to make possible applications where vast numbers of devices coexist on a communication network. A typical example is a wireless network comprising low-cost sensors that forward measurements from their locations to a fusion center. Given the variety of applications supported by, and the ubiquitous nature of these networks, for IoT solutions to be viable, the embedded IoT devices will have to cost very little (less than $1, according to some estimates [2]). Such devices will be resource challenged, possess very limited communication and computing capabilities and support little memory. Control policies for such networks will, perforce, need to be simple and not result in excessive overheads.

These constraints are starkly different from those encountered in the transmission of traditional voice and packet data over wireline or cellular wireless networks. In cellular systems, for example, most scheduling decisions come from the base station and hence, control is centralized [3, Chap. 6, 13], but some Quality of Service (QoS) is expected – low packet delay, for instance [4], [5]. In contention access systems such as WiFi, scheduling is distributed but service is best effort [6] or, at best, differentiated – such as with the "Enhanced Distributed Channel Access" [7] and the "Enhanced Distributed Coordination Function" (E-DCF) mechanisms in the IEEE 802.11e standard [8]. In many IoT applications, e.g., condition monitoring or predictive maintenance, there is a need for low overhead distributed scheduling (for the earlier reasons) while also providing QoS.

Consequently, resource allocation techniques developed to handle services such as file transfer and packet-voice might not be appropriate for wireless networks of resource challenged nodes, that need to provide QoS to the applications they are carrying. Most existing medium access protocols and scheduling algorithms suffer from limitations – the WiFi protocol and its Differentiated Services version are simple, but do not really provide any guarantees. Recent standards for the IoT, such as 6TiSCH on the other hand, come with guarantees but require a lot of information exchange and central coordination [9], [10]. Our aim in this paper is to propose decentralized Medium Access Control (MAC) protocols with a focus on low packet delay (i.e., latency) and reduced exchange of control information across the network.

The traditional approach for dynamic resource allocation has been to use backlog or queue length information to

opportunistically schedule transmissions. One of the seminal contributions to scheduling in constrained queueing systems is the work of Tassiulas and Ephremides [11]. These authors modeled a wireless network as a network of queues with pair-wise scheduling constraints (corresponding to wireless interference, half-duplex operation, etc.), and several flows over the network, each with its ingress queue and egress queue. A time-slotted model was considered in [11], with global queue scheduling decisions needing to be made at the beginning of every slot. Scheduling constraints (such as link interference or half-duplex constraints) were modelled by pair-wise scheduling constraints, represented by an *interference graph*; queues adjacent in the interference graph cannot be scheduled in the same slot. With stochastic arrivals to each flow to be routed from their ingress to egress points, the authors develop MaxWeight a *centralized* scheduling algorithm which requires the queue lengths of all nodes in every scheduling slot and show that it is throughput-optimal, i.e., it stochastically stabilizes all queues under any stabilizable arrival rate.

Attempts to decentralize MaxWeight include approximations based on message passing between nodes [12], [13], or using queue lengths to modulate backoff parameters in Carrier Sense Multiple Access (CSMA) and ALOHA [14], [15]. Both of these methods, while being throughput-optimal, suffer from poor delay performance. Another method to reduce the amount of information required for scheduling is proposed in [1], where, for two classes of constrained queueing systems, algorithms relying only on the empty-nonempty state of queues is proposed and analyzed for delay performance. Our interest lies in the second half of [1], wherein a scheduling algorithm is proposed for a system of N parallel queues in which adjacent queues cannot be served simultaneously. For such a network, the authors provide a technique to improve the delay performance of a *given* scheduling policy (assuming one exists). For $N = 3$, the authors provide a policy that is mean delay optimal over the entire stability region. For $N = 4$, Ji et al. [15] provide a policy that heavy-traffic delay optimal. It is not yet clear if it is possible to extend these algorithms to general wireless networks while preserving performance guarantees such as throughput-optimality. In Sec. I-A below, we explain our contributions in greater detail.

A. Our Contributions and Organization

We consider a system of N wireless links (transmitter-receiver pairs). Time is slotted and each transmitter has an independent arrival process of packets embedded at slot boundaries. Only one packet can be transmitted across a link in any slot. There are scheduling constraints (link activation constraints) that constrain which links can simultaneously transmit in any slot. Packets that arrive and cannot be immediately transmitted, wait in a queue at the transmitter[1].

When the links are "collocated," i.e., all links interfere with each other, only one link can be activated in any slot. In a collocated network, for stabilizable arrival rates (see Sec. II), it is known that any policy that transmits a packet from any nonempty queue is not only stabilizing, but is delay optimal in the strong sense of stochastically minimising the sum-queues process (see Defn. IV-A).

We begin with the system model in Sec. II, wherein we describe the two classes of interference networks considered in this article: "path-graph networks" (Sec. II-B) and "cluster-of-cliques" networks (Sec. II-C). We restrict our study to two subclasses of scheduling policies: Maximum Size Matching (MSM) policies (Sec. III-A) that serve a set of links with the largest number of nonempty non-conflicting queues in every slot, and Queue Nonemptiness based (QNB) policies that use only the empty-nonempty statuses of network queues for scheduling (Sec. III-B).

We then provide a complete characterization of the set of QNB-MSM policies for path-graph networks for the case with $N = 3$ queues (Sec. IV). The fact that the policies we discuss do not require any information about the queues except their empty-nonempty status helps satisfy our reduced state information requirement. We establish several interesting results about (in)stability and delay optimality. Specifically, for $N = 3$, we supply a formal proof that there is a delay optimal policy in the QNB-MSM class of policies, an observation that had been made in [1]. We find that, for $N = 3$, one of the QNB-MSM policies is not even throughput optimal, thus bringing forth the need for careful design of the scheduler.

In our work, we provide a study of QNB-MSM scheduling policies (explicitly constructing several such policies along the way), and show how scheduling policies for larger networks can be constructed by the novel method of policy *splicing*. Continuing with path-graph networks, we propose a "policy splicing" technique (see Figures 3 and 4) to combine policies for small networks to construct low delay policies for larger networks (Sections V and V-D). We use this technique to propose QNB-MSM scheduling policies for several such networks. We also provide an in-depth analysis of delay (Sec. VI), culminating in a result that shows that there do not exist delay optimal QNB-MSM policies for such networks with $N \geq 4$ queues (Thm. 12).

We then extend our theory of MSM policies to schedule transmissions over cluster-of-cliques constraint networks (Sec. VII) and also discuss multiple methods to further reduce the amount of state information that has to be exchanged across the network to make these protocols amenable to distributed implementation. We finally use this theory to propose a throughput-optimal protocol, akin to the QZMAC protocol [16], wherein scheduling decisions are taken using only the information about activity on the channel (or lack thereof) that can be *sensed* by the nodes and will study its performance in detail (Sec. VIII). We then present numerical results (Sec. IX) showing the performance of our proposed policies, and comparisons with standard, high-overhead state-based policies such as the MaxWeight-α family [17]. These simulation studies show that MaxWeight and MaxWeight-α, while guaranteeing queue stability, can perform poorly in delay performance in comparison with QNB-MSM policies.

[1]Activating a link, therefore, is the same as serving its associated queue. Henceforth, we will use the terms "link" and "queue" interchangeably.

II. The Scheduling Problem: Models and Notation

In Sec. II-A, we describe the general network model, and specify the optimal scheduling problem in Sec. II-A1. Then, in Sec. II-B and Sec. II-C we restrict the general model to the cases that we provide results for in the remainder of the paper.

There are several interfering links (*transmitter-receiver pairs*), where each transmitting node has a stream of arriving packets. Time is slotted, and all links are synchronized to the time slots. In each slot, each scheduled link can transmit one packet. Packets that are not transmitted remain in the queues. Thus, we have a discrete time queue scheduling problem that belongs to the general class introduced in [11]. Note, from the preceding discussion, that activating a link in a time slot is the same as serving its associated queue.

A. The General Queue Scheduling Model

We consider a system comprising N queues, where, as mentioned before, each queue models a radio link in a wireless network. The leading edges of time slots are indexed $0, 1, 2, \cdots$. Exogenous arrivals to the queues are embedded *at* slot boundaries, $t = 0, 1, 2, \cdots$, with the number of packets arriving to Queue i at time t being denoted by the random variable $A_i(t)$. $A_i(t)$ is assumed IID[2] across time and independent across queues and is modelled as a Bernoulli random variable with mean λ_i i.e., $P(A_i(t) = 1) = \lambda_i$, $\forall t \geq 1$. However, we will remove this restriction to include batch IID arrivals in Sec. VII. We use $\mathbf{Q}(t) = [Q_1(t), \ldots, Q_N(t)]$ to denote the vector of all queue lengths at time t. The queue length process is embedded at the beginnings of time slots, so $Q_i(t)$, $\forall t \geq 0$, is measured at $t+$, i.e., just *after* the arrival. The duration of a slot is assumed to include packet transmission time, the receive-transmit turn around time at the receiver, the MAC layer acknowledgement (ACK) time[3], and any scheduling overhead. Packet transmissions are assumed to take exactly one time slot and succeed with probability[4] 1. The random variable, $D_i(t)$, indicating the departure of a packet from Queue i at time t, is such that $D_i(t) = 1$ if and only if Queue i is scheduled in slot t *and* $Q_i(t) > 0$, else $D_i(t) = 0$; here, the departure is assumed to end just before the leading edge of slot $(t + 1)$, i.e., at $(t + 1)-$.

The *offered service* process to Queue i, $\{S_i(t), t \geq 0\}$, is defined as follows: $S_i(t) = 1$ whenever Queue i is given access to the channel, so that $D_i(t) = S_i(t)\mathbb{I}_{\{Q_i(t)>0\}}$, $\forall t \geq 0$, $1 \leq i \leq N$. Depending on the interference constraints, it may be possible to serve only a subset of queues in a given slot. For example, (2) gives the constraints for path-graph interference networks and (4) for Star-of-Cliques networks. The vector $\mathbf{S}(t) := [S_1(t), \ldots, S_N(t)]$ satisfying the interference constraints is called an *activation vector*. Thus, for every queue i,

$$\begin{aligned} Q_i(t+1) &= Q_i(t) - D_i(t) + A_i(t+1) \\ &= (Q_i(t) - S_i(t))^+ + A_i(t+1), \ \forall t \geq 0. \end{aligned}$$

[2]"IID" stands for *independent and identically distributed.*

[3]Most wireless systems require a MAC layer acknowledgement to combat high high packet error rates

[4]The effects of fading and channel errors are not considered here and are a subject of future research.

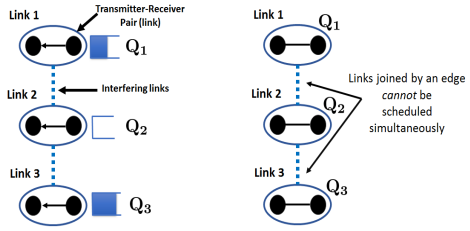


Fig. 1: The basic path-graph interference system with $N = 3$ communication links along with the associated packet queues (left) and its conflict graph (right). The interference constraints are such that physically adjacent queues cannot be served simultaneously.

Denote by $\boldsymbol{\zeta}(t) := [\mathbb{I}_{\{Q_1(t)>0\}}, \ldots, \mathbb{I}_{\{Q_N(t)>0\}}]$ the system's *occupancy vector* at time t, i.e., the empty-nonempty state of each of the N queues. Let $\mathcal{V} \subset \{0, 1\}^N$ be the set of all activation vectors. A scheduling policy $\pi := \{\mu_0, \mu_1, \ldots\}$ decides which queues are allowed to transmit in each slot as a function of the available history $\mathcal{H}_t$, which comprises the past states and actions known to the controller, and the current (known) queue state. Specifically, $\mu_t : \mathcal{H}_t \to V$ is an $N \times 1$ vector, and $S_i(t) = \mu_t(i)$. When the schedule depends only on state and not on time, the resulting policies are of the form $\pi = \{\mu, \mu, \ldots\}$, and are said to be *stationary.* We will focus on stationary policies in this article.

1) Performance Metric: By *stability* of the process $\{\mathbf{Q}(t), t \geq 0\}$ we will mean that

$$\limsup_{T\to\infty} \frac{1}{T} \sum_{t=0}^{T-1} \sum_{i=1}^{N} \mathbb{E}^{(\pi)}_{\mathbf{Q}(0)} Q_i(t) < \infty. \tag{1}$$

This condition is commonly known as *strong stability* [18]. A policy that ensures (1) is said to be *stabilizing,* and an arrival rate vector for which a stabilizing policy exists is said to be *stabilizable*. The closure of the set of all stabilizable rate vectors is called the *throughput capacity region* of the network [11], and a policy that is stabilizing for every arrival rate vector in the interior of this region is called *throughput-optimal* (T.O.). The set of arrival rates that are stabilizable under a given fixed policy is called *stability region* of the policy.

B. Path Graph interference networks

The first system we will study in the subsequent sections is modelled by N parallel queues (see Fig. 1). The scheduling constraints are the same as the second model in Tassiulas and Ephremides 1994 [1], namely that Queue i and Queue $i + 1$ cannot be served simultaneously for $1 \leq i \leq N - 1$. These interference constraints enforce the following rule on the offered service process $\mathbf{S}(t)$, $\forall t \geq 0$

$$S_i(t) + S_{i+1}(t) \leq 1, \ \forall t \geq 0, 1 \leq i \leq N - 1. \tag{2}$$

The conflict graph associated with the system is a called *path graph* [19], [20]. Standard analysis [11] show that the capacity region of this network is the set

$$\Lambda_N := \left\{\boldsymbol{\lambda} \in \mathbb{R}^N_+ \mid \lambda_i + \lambda_{i+1} \leq 1, \ \forall 1 \leq i \leq N - 1\right\}, \tag{3}$$

whose interior, Λ^o_N, is the set of all stabilizable rate vectors.

C. The Cluster-of-Cliques (CoC) graph networks

In the remainder of the paper, we will refer to the conflict graph associated with a collocated network, i.e., a fully connected graph or subgraph, as a *clique*. The system under consideration comprises multiple cliques and the exact nature of the interference relations *across* cliques are described in detail below. The number of packets arriving to Queue j in Clique i at time t is denoted by the random variable $A_{i,j}(t)$. As before, $Q_{i,j}(t)$, the backlog of Queue j in Clique i is measured at $t+$, $t \geq 0$, i.e., just *after* the arrival. Once again, as before, for every (i, j),

$$\begin{aligned} Q_{i,j}(t+1) &= Q_{i,j}(t) - D_{i,j}(t) + A_{i,j}(t+1) \\ &= (Q_{i,j}(t) - S_{i,j}(t))^+ + A_{i,j}(t+1), \ \forall t \geq 0. \end{aligned}$$

Depending on the underlying conflict graph, the CoC networks studied in this paper are broadly classified into two classes

Star-of-Cliques networks (SoC): Consider an interference graph consisting of a central fully-connected subgraph (central clique) surrounded by $N-1$ cliques (see Fig. 2b). In other words, the network's conflict graph consists of N cliques denoted $C_1, \ldots, C_N$, and clique C_i consists of $\mathcal{N}_i$ vertices – an *arbitrary number* of cliques each having *arbitrarily many* communication links (queues). Transmissions in C_1 interfere with those in all other cliques while the transmissions in C_i, $i \geq 2$ interfere with those in C_1 only. Coming to the offered service processes, for any two queues $Q_{i,j}$ and $Q_{k,l}$ in the system, the interference constraints enforce the rule

$$S_{i,j}(t) + S_{k,l}(t) \leq 1, \ \forall t \geq 0, \text{ if } i = k, \text{ or } i = 1, \text{ or } k = 1. \quad (4)$$

Let $\mathcal{N} \equiv \sum_{i=1}^{N} \mathcal{N}_i$ denote the total number of queues in the system. The capacity region of this system is given by

$$\begin{aligned} \Lambda_s^{(N)} := \Bigg\{ \boldsymbol{\lambda} \in \mathbb{R}_+^{\mathcal{N}} \Bigg| & \sum_{j=1}^{\mathcal{N}_1} \lambda_{1,j} \\ & + \sum_{k=1}^{\mathcal{N}_i} \lambda_{i,k} \leq 1, \ i \in \{2, \cdots, N\} \Bigg\} \end{aligned} \quad (5)$$

(the subscript s highlights the fact that this is the Star-of-Cliques model).

Linear-Array-of-Cliques (LAoC): This system consists of N cliques $C_1, C_2, \cdots, C_N$, but unlike the SoC model, all transmissions in C_{i-1} interfere with those in C_i, $i \in \{2, \cdots, N\}$ and vice-versa (see Fig. 2a). As in the SoC model, Clique C_i comprises $\mathcal{N}_i$ queues and $\mathcal{N} \equiv \sum_{i=1}^{N} \mathcal{N}_i$ denotes the total number of queues in the system. Since transmissions in adjacent cliques interfere with each other, for the offered service processes of any two queues $Q_{i,j}$ and $Q_{k,l}$ in the system, we have

$$S_{i,j}(t) + S_{k,l}(t) \leq 1, \ \forall t \geq 0, \text{ if } k = i+1, \ \forall 1 \leq i \leq N-1. \quad (6)$$

The capacity region of this system is given by

$$\begin{aligned} \Lambda_l^{(N)} := \Bigg\{ \boldsymbol{\lambda} \in \mathbb{R}_+^{\mathcal{N}} \Bigg| & \sum_{j=1}^{\mathcal{N}_i} \lambda_{i,j} \\ & + \sum_{k=1}^{\mathcal{N}_{i+1}} \lambda_{i+1,k} \leq 1, \ i \in \{1, \cdots, N-1\} \Bigg\} \end{aligned} \quad (7)$$

(the subscript l highlights the fact that this is the Linear-Array-of-Cliques model). As before, the vector $\mathbf{S}(t) := [S_1(t), \ldots, S_{\mathcal{N}}(t)] \in \{0,1\}^{\mathcal{N}}$ is called an *activation vector* if it satisfies the constraints in (4) and (6) in the SoC and LAoC systems, respectively. We now begin our study with path graph interference networks.

III. Scheduling in Path Graph Models

A. Maximum Size Matching (MSM) Policies

The sets of transmitters and receivers can be viewed as nodes in a bipartite (communication) graph, whose edges are the links between these transmitters and receivers, see Fig. 1. The link activation constraints are superimposed on the bipartite communication graph. With this structure in mind, and to conform to standard terminology from bipartitite graphs [21], the following definition holds.

Definition. A policy π is a *Maximum Size Matching (MSM)* policy if in every slot the policy schedules the maximum number of links with nonempty transmitter queues subject to the interference constraints.

For example, in a path graph network with $N = 7$ queues, if $\mathbf{Q}(t) = [1, 2, 0, 0, 4, 3, 3]$, a policy that schedules queues $1, 5$ and 7 or $2, 5$ and 7 is MSM while a policy that schedules queues $1, 7$ only, is not MSM. It might be expected that the policy must schedule as many queues as possible to maximise throughput and minimise delay. Indeed, it is shown in [1] that any policy defined on such path-graph networks can be improved into an MSM policy that will provide stochastically better delay. Interestingly, we show later that even non-MSM policies can be stabilising.

In this paper, we will refer to Queues 1 and N in a path graph as the "outer" queues and the other $N-2$ queues as the "inner" queues. The inner queues are, in a sense, more constrained for scheduling, since they cannot be served in any slot in which either adjacent queue is scheduled, while service to Queue 1 depends only on whether Queue 2 is being served, which makes it less constrained from the perspective of service.

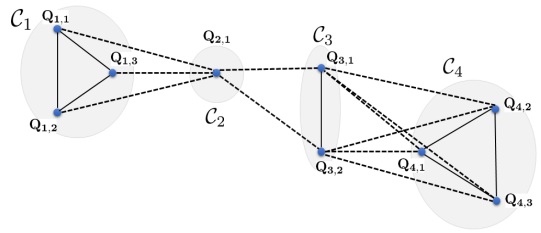
Lem. 4.1, in [1], defines a class of policies that is more restrictive than MSM that can be described informally and succinctly as follows.

1) the policy should be MSM, and
2) the policy should prioritize *inner* queues over *outer* queues while breaking ties.

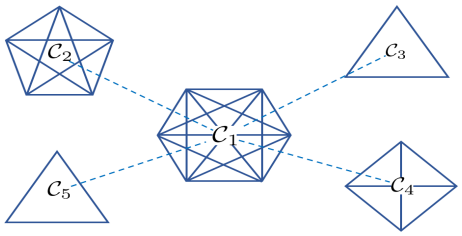
Specifically, the authors provide a sufficient but *not necessary* condition for an activation vector to serve the largest number of nonempty queues in a slot. Recall that $\boldsymbol{\zeta}(t) = [\mathbb{I}_{\{Q_1(t)>0\}}, \ldots, \mathbb{I}_{\{Q_N(t)>0\}}]$ and define $\mathcal{U}(\boldsymbol{\zeta}(t)) \subset \mathcal{V}$ as the set of all activation vectors that serve the largest number of queues in slot t when the occupancy vector is $\boldsymbol{\zeta}(t)$. With this, one simply needs to ensure that in every slot, the policy chooses activation vectors only from $\mathcal{U}(\boldsymbol{\zeta})$, to ensure that it is MSM. In fact, we will show that in several interference graphs, $\boldsymbol{\zeta}(t)$ is sufficient not only for stability but also for delay optimality.

Notation: Classes of scheduling policies

- $\Pi^{(N)}$: the class of all policies.



(a) The conflict graph associated with a Linear-Array-of-Cliques (LAoC) network. While this is clearly *neither* fully connected nor a path-graph network, we will show how to extend ideas from the analysis of path-graph networks to construct scheduling protocols for such networks.



(b) The conflict graph associated with a Star-of-Cliques (SoC) network. A dotted line connecting cliques C_i and C_j means that transmissions in the two cliques cannot take place simultaneously.

Fig. 2: Cluster-of-Cliques networks.

- $\Gamma_M^{(N)}$: the class of all MSM policies.
- $\Pi_M^{(N)}$: the class of all policies that take only the occupancy vector $\boldsymbol{\zeta}(t)$ as input and activate the largest number of non empty queues in every slot, i.e., MSM policies that require only the empty or nonempty status of the queues in the network. We will call the policies that use only $\boldsymbol{\zeta}(t)$ Queue Non-emptiness Based (QNB) policies; see Sec. III-B.
- $\tilde{\Pi}^{(N)}$: the class of all MSM policies within $\Pi_M^{(N)}$ that additionally break ties in favour of inner queues (see condition 2 earlier in this subsection).

Note that $\Pi^{(N)} \supsetneq \Gamma_M^{(N)} \supsetneq \Pi_M^{(N)} \supsetneq \tilde{\Pi}^{(N)}$. Going back to our 7-queue example, when $\boldsymbol{\zeta}(t) = [1,1,0,0,1,1,1]$, policies that choose $\mathbf{S}(t) = [1,0,0,0,1,0,1]$ can be in $\Pi_M^{(7)}$ but not in $\tilde{\Pi}^{(7)}$, while those that choose $\mathbf{S}(t) = [0,1,0,0,1,0,1]$ can be in $\tilde{\Pi}^{(7)}$.

B. Queue Nonemptiness-Based (QNB) Scheduling

In the previous subsection we defined "queue nonemptiness based" policies, i.e., those that require only the knowledge of the occupancy vector, $\boldsymbol{\zeta}(t)$. Clearly, this contains much less information than the vector $\mathbf{Q}(t)$ that MaxWeight requires. While $\boldsymbol{\zeta}(t)$ needs only N bits per slot for encoding, $\mathbf{Q}(t)$ may require an arbitrarily large number of bits per queue per slot, depending on the buffer size and quantization used. So now, $\pi = \{\mu, \mu, \cdots\}$ with $\mu : \{0,1\}^N \to \mathcal{V} \subsetneq \{0,1\}^N$, the set of all activation vectors.

Although it is well-known that fully-connected interference graphs admit throughput-optimal, queue nonemptiness-based scheduling algorithms, (e.g., schedule any nonempty queue) it is not immediately clear how to stabilize other interference graphs with reduced state policies. Moreover, it is not clear if using a reduced state scheduler automatically ensures delay optimality, since even MaxWeight, which uses complete knowledge of $\mathbf{Q}(t)$ in every slot, is only known to be asymptotically delay optimal in such networks [22].

We now provide a key sufficient condition for a scheduling policy, that guarantees throughput-optimality. This result will form the basis for constructing strongly stable policies that use only $\{\boldsymbol{\zeta}(t),\ t \geq 0\}$, throughout the remainder of the paper.

Lemma 1. Consider the class of systems described in Sec. II-B, and define property $\mathcal{P}$ as

$$D_i(t) + D_{i+1}(t) = 0 \iff Q_i(t) + Q_{i+1}(t) = 0, \qquad (\mathcal{P})$$

for all $t \geq 0$, and for $1 \leq i \leq N-1$. Any policy that satisfies property $\mathcal{P}$ in every slot t, is throughput-optimal.

Remark. Note that condition $(\mathcal{P})$ depends only on the reduced state $\boldsymbol{\zeta}(t)$. In words, $(\mathcal{P})$ reads: "for a pair of neighboring queues, there is no departure from either of these queues if and only if both the queues are empty." One direction is clear: when both queues are empty there can be no departures. For example, with $N = 4$ and $\boldsymbol{\zeta}(t) = (1,1,1,1)$, $\mathbf{S}(t) = (1,0,1,0)$ satisfies condition $(\mathcal{P})$, but $\mathbf{S}(t) = (1,0,0,1)$ does not.
Also note that extensions of this property for the cluster-of-cliques system will be discussed and presented in Sec. VII, when we take up a detailed study of these conflict graphs.

PROOF SKETCH. The proof of Lem. 1 is based on a novel Lyapunov function $L(t) : \mathbb{N}^N \to \mathbb{R}_+$, defined as

$$L(\mathbf{Q}(t)) := \sum_{i=1}^{N-1} (Q_i(t) + Q_{i+1}(t))^2. \qquad (8)$$

Instead of showing negative drift per queue state, as is common in the analysis of several MaxWeight-style algorithms, we provide an averaging argument to show overall negative drift of the Lyapunov function, and appeal to the telescoping sum technique to prove strong stability. The proof is deferred to Sec. XI-D in the supplementary material. ■

IV. Path Graph Conflict Model with $N = 3$: QNB Scheduling

In this section, we first completely characterize $\Pi_M^{(3)}$ and the subclass $\tilde{\Pi}^{(3)}$, and explore stability and delay optimality for this system. This study will provide some insights into the nature of MSM policies in general and, more importantly, in this process, the policies we propose here will act as *building blocks* for policies for larger-N systems. Before we embark on this analysis, we would like to make a few preliminary observations about $\Pi^{(3)}$. For the reader's convenience a glossary of notation is provided in the supplementary material: XI-B.

Note that with 3 queues, in any given slot t, a policy can choose either $\mathbf{S}(t) = [1,0,1]$ which serves Queues 1 and 3, or $[0,1,0]$ which serves Queue 2. So, a queue nonemptiness-based policy maps every state vector $\boldsymbol{\zeta}(t)$, of which there are 8 alternatives, to one of these two activation vectors, giving us $2^8 = 256$ QNB policies in all. Suppose $|\ A\ |$ denotes the cardinality of set A. It is easily shown that upon imposing the MSM condition, this number reduces to 4, i.e., $|\Pi_M^{(3)}| = 4$ in our techreport [23, Sec. V].

Depending on the mapping from $\boldsymbol{\zeta}(t)$ to the activation vector, we denote the 4 MSM policies $\pi_{TD}^{(3)}, \pi_{BU}^{(3)}, \tilde{\pi}_{IQ}^{(3)}, \pi_{OQ}^{(3)}$. We will follow the scheme below in the remainder of the paper.

- The subscripts "TD" and "BU" stand for "Top-Down" and "Bottom-Up," respectively and the reason for this nomenclature will become apparent shortly.
- A "$\sim$" in the superscript always represents a policy in $\tilde{\Pi}^N$, regardless of any subscripts. It indicates that these policies always break ties in favor of inner queues. For example, in Sec. IV-A, $\tilde{\pi}^{(3)}_{IQ} \in \tilde{\Pi}^3$.

TABLE I: Comparison of $\mathbf{S}(t)$ under $\pi^{(3)}_{TD}$, $\pi^{(3)}_{BU}$, $\tilde{\pi}^{(3)}_{IQ}$ and $\pi^{(3)}_{OQ}$

$\boldsymbol{\zeta} = [\zeta_1(t), \zeta_2(t), \zeta_3(t)]$	$\mathbf{S}(t)$			
	$\pi^{(3)}_{TD}$	$\pi^{(3)}_{BU}$	$\tilde{\pi}^{(3)}_{IQ}$	$\pi^{(3)}_{OQ}$
000	101	101	101	101
001	101	101	101	101
010	010	010	010	010
011	010	101	010	101
100	101	101	101	101
101	101	101	101	101
110	101	010	010	101
111	101	101	101	101

The complete descriptions of all these policies are given in Table. I. Let us consider each of these policies in turn. In what follows we will describe and analyze each of these policies in detail. Notice from the entries corresponding to the rows $\boldsymbol{\zeta} = [011]$ and $\boldsymbol{\zeta} = [110]$ that $\pi^{(3)}_{TD}$ and $\pi^{(3)}_{BU}$ are complementary policies, and so are $\tilde{\pi}^{(3)}_{IQ}$ and $\pi^{(3)}_{OQ}$. Specifically, each of these four policies induces the following priority order, which will become clear when we consider each of them individually later:

- $\pi^{(3)}_{TD}$ gives *decreasing* priority to Queues 1, 2 and 3 in that order. This policy clearly gives absolute priority to Queue 1, i.e., serves Queue 1 whenever it is nonempty. Hence, the subscript "TD," since this policy, in a sense, establishes a "Top-Down" priority (thinking of Queue 1 as the "top," and Queue 3 as the "bottom"),
- $\pi^{(3)}_{BU}$ gives *increasing* priority to Queues 1, 2 and 3 in that order,
- $\tilde{\pi}^{(3)}_{IQ}$ gives maximum priority to Queue 2 the *inner* queue (check rows in Table. I corresponding to $\boldsymbol{\zeta} = [011]$ and $\boldsymbol{\zeta} = [110]$), while not compromising the MSM property. This is, of course consistent with the fact that it lies in the $\tilde{\Pi}^{(3)}$ class where ties are always broken in favor of inner queues, and
- $\pi^{(3)}_{OQ}$ prioritizes the two *outer* queues.

To begin with, we show that $\pi^{(3)}_{TD}$ and $\pi^{(3)}_{BU}$ are T.O. Both these policies will later be used as building blocks to construct T.O. policies for larger systems and are therefore very important to our study.

Theorem 2. $\pi^{(3)}_{TD}$ and $\pi^{(3)}_{BU}$ are both throughput-optimal.

PROOF SKETCH. The proof of Thm. 2 uses the fact that under $\pi^{(3)}_{TD}$, Queues 1 and 2 form a priority queueing system and are stable. We then show that Queue 3 is served "sufficiently often" to ensure stability. $\pi^{(3)}_{BU}$ simply swaps the priorities of Queues 1 and 3 and its proof proceeds mutatis mutandis. The complete proof is available in Sec. XI-E. ■

The rest of this section is organized as follows. In Sec. IV-A we study the stability and delay performance of $\tilde{\pi}^{(3)}_{IQ}$. We then analyze the stability of $\pi^{(3)}_{OQ}$ in Sec. IV-B and show that it is, in fact, *unstable*. We then study a non-MSM policy $\pi^{(3)}_{IQ}$, and show it to be T.O. This policy is used as a building block in policies for larger path graph systems later. Finally, in Sec. IV-D we study a version of the "Flow-in-the-Middle" problem, an interesting phenomenon commonly observed in contention access networks, show that path graph networks also exhibit this behavior and analyze its consequences.

A. Analysis of $\tilde{\pi}^{(3)}_{IQ}$

This policy can be restated as follows.

At time t:

1) *If $Q_1(t) > 0$ and $Q_3(t) > 0$, then* $\mathbf{S}(t) = [1, 0, 1]$.
2) *Else, if $Q_2(t) > 0$, then* $\mathbf{S}(t) = [0, 1, 0]$.
3) *Else* $\mathbf{S}(t) = [1, 0, 1]$.

We begin analyzing the policy by proving that it is Throughput Optimal.

Theorem 3. $\tilde{\pi}^{(3)}_{IQ}$ is throughput-optimal.

PROOF SKETCH. The proof of this result involves showing that $\tilde{\pi}^{(3)}_{IQ}$ satisfies $(\mathcal{P})$ in Lem. 1 and is therefore throughput-optimal. The proof is available in Sec. XI-F. ■

We next turn to the delay performance of the policy $\tilde{\pi}^{(3)}_{IQ}$. Thm. 4.2 in [1] defines a projection operator $L : \Pi^{(N)} \to \Gamma^{(N)}_M$ that takes any policy $\pi \in \Pi^{(N)}$ and produces an MSM policy, $L(\pi)$. It is then shown that the sum queue length with this MSM policy $L(\pi)$ is stochastically smaller than with π. Specifically, if $\mathbf{Q}^{\pi}(t)$ denotes the backlog induced by some policy π, then Thm. 4.2 in [1] shows that when the systems upon which π and $L(\pi)$ act are started out in the same initial state and the arrivals have the same statistics, then

$$\sum_{i=1}^{N} Q_i^{L(\pi)}(t) \overset{st}{\leq} \sum_{i=1}^{N} Q_i^{\pi}(t), \ \forall t \geq 0, \tag{9}$$

where *st* denotes stochastic ordering. Notice that in the above stochastic ordering relation is required to hold for any arrival rate vector in the system's capacity region. Extending this gives rise to the concept of a *Uniformly Delay Optimal* Policy (in the literature, this is also referred to as *Sample-Path Optimality* [24], [25]):

Definition. For a path graph interference network with N queues, a policy $\pi^* \in \Pi^{(N)}$ is said to be *Uniformly Delay Optimal* if, given *any* policy $\pi \in \Pi^{(N)}$, when the systems upon which π and π^* act are started out in the same initial state and with the same arrivals statistics and for every arrival rate $\boldsymbol{\lambda} \in \Lambda_N$,

$$\sum_{i=1}^{N} Q_i^{\pi^*}(t) \overset{st}{\leq} \sum_{i=1}^{N} Q_i^{\pi}(t), \ \forall t \geq 0. \tag{10}$$

In [1, Remark 2, pp. 353], it is stated that for $N = 3$ there is exactly one MSM policy and that, as a result of Thm. 4.2 therein, is also sum queue length optimal (in the stochastic

ordering sense). It is clear, however, that for $N = 3$ there are 4 MSM policies. Indeed, the unique MSM policy that the authors refer to in [1] is $\tilde{\pi}^{(3)}_{IQ}$, which also prioritises inner queues. However, the projection operator $L(\cdot)$ does not ensure inner queue prioritisation i.e., does not ensure condition 3 in Lem. 4.1 therein. Thus using Theorem 4.2 in [1], in the present context, we can only conclude that that any one of the MSM policies could be delay optimal. It requires a further step in the proof to show that $\tilde{\pi}^{(3)}_{IQ}$ is, indeed, the unique uniformly delay optimal policy for $N = 3$. We proceed to do so below.

Theorem 4. For any policy $\pi \in \Pi^{(3)}$, let the system backlog vector at time t be denoted by $\mathbf{Q}^{\pi}(t)$ and the backlog with $\tilde{\pi}^{(3)}_{IQ}$ be denoted by $\mathbf{Q}^{\tilde{\pi}^{(3)}_{IQ}}(t)$. Also let $\mathbf{Q}^{\pi}(0) = \mathbf{Q}^{\tilde{\pi}^{(3)}_{IQ}}(0)$. Then,

$$\sum_{i=1}^{3} Q_i^{\tilde{\pi}^{(3)}_{IQ}}(t) \overset{st}{\leq} \sum_{i=1}^{3} Q_i^{\pi}(t), \ \forall t \geq 0, \tag{11}$$

where "*st*" denotes stochastic ordering.

PROOF SKETCH. The proof technique is essentially the same as that of Thm. 4.2 in [1], except that we make the observation that a key step in that proof has more general applicability. It involves constructing a sequence of policies each of which shows better delay than its predecessor and than a general policy π. The limit of this sequence of policies is then shown to uniquely be $\tilde{\pi}^{(3)}_{IQ}$. The proof is deferred to Sec. XI-G. ■

Remark. Another important consequence of the operator L satisfying (9) is that whenever a given policy π is throughput-optimal, so is $L(\pi)$. Please refer [23, Sec. V] for details. Directly analyzing the stability and delay properties of the policies we propose in the sequel is very difficult. We therefore develop indirect methods to analyze them by first analyzing nonMSM policies whose behavior can be understood easily, but that do not show desirable delay properties and study the proposed policies as modifications (such as projection) of these simpler policies, with the modifications giving rise to better delay performance.

B. Analysis of $\pi^{(3)}_{OQ}$

This policy prioritizes the outer queues and can be restated as follows.

At time t:

1) *If either* $Q_1(t) > 0$ *or* $Q_3(t) > 0$, *then* $\mathbf{S}(t) = [1, 0, 1]$.
2) *Else* $\mathbf{S}(t) = [0, 1, 0]$.

It turns out, analogous to the observation by McKeown et al [26] that this MSM policy is, in fact, not throughput-optimal.

Proposition 5 (*MSM but not throughput-optimal*)**.** $\pi^{(3)}_{OQ}$ is *not* throughput-optimal.

The proof of this result involves constructing an arrival rate vector for which the offered service rate to one of the queues is strictly smaller than the arrival rate. It is available in Sec. XI-H of the Appendix. Once again, this proof technique is important and we will repeatedly use it in the sequel.

This completes the characterization of $\Pi^{(3)}_{M}$.

C. Policies outside $\Pi^{(3)}_{M}$

We now propose and analyze a policy that we denote $\pi^{(3)}_{IQ}$, and show the rather surprising result that *it is throughput-optimal despite not being MSM*. This stability comes from the fact the policy prioritizes the inner queue. However, since it is not MSM, its delay performance is not very good (see the simulation results in Sec. IX). This policy will become important shortly as a fundamental building block while constructing policies for larger systems using a novel *Policy Splicing* technique.

At time t

1) *If* $Q_2(t) > 0$, *then* $\mathbf{S}(t) = [0, 1, 0]$,
2) *Else* $\mathbf{S}(t) = [1, 0, 1]$.

Since $\boldsymbol{\zeta}(t) = [1, 1, 1] \mapsto [0, 1, 0]$, this policy is not MSM. However, we have

Proposition 6 (*A non-MSM but throughput-optimal policy*)**.** $\pi^{(3)}_{IQ}$ is throughput-optimal.

PROOF. The key tool behind the proof of this result is the throughput-optimality Lem. 1. It is easily checked that $\pi^{(3)}_{IQ}$ satisfies $(\mathcal{P})$ in every slot and thus, by Lem. 1, is throughput-optimal. ■

D. A Randomized Policy: The Flow-in-the-Middle Problem

The "Flow-in-the-Middle" problem, or FIM for short, is a practical problem faced by all networks that employ CSMA at the MAC layer. It refers to the situation in which, the flow (or link) within the interference range of two adjacent flows, i.e., in the middle (Queue 2, in our model), can remain starved of service for long periods of time in the presence of uncoordinated transmissions. This problem has been studied in detail both analytically and experimentally in asynchronous *continuous-time* systems in the literature [27]–[32]. In this section, we aim to model such a scenario, albeit in slotted time, and understand whether such a phenomenon can occur in the network under study, which naturally leads to the central link (or flow) being starved for extended periods of time. Recall that the occupancy vector is defined as $\boldsymbol{\zeta}(t) := \left[\mathbb{I}_{\{Q_1(t)>0\}}, \mathbb{I}_{\{Q_2(t)>0\}}, \mathbb{I}_{\{Q_3(t)>0\}}\right]$. Consider the policy $\rho^{(3)}_{\gamma}$ indexed[5] by a randomization parameter $\gamma \in [0, 1]$ defined as follows.

At time t:

- *If* $\boldsymbol{\zeta}(t) = [1, 1, 1]$ *or* $[1, 0, 1]$, *then* $\mathbf{S}(t) = [1, 0, 1]$.
- *Else, if* $\boldsymbol{\zeta}(t) = [1, 1, 0]$ *or* $[0, 1, 1]$, *then*
  1) $\mathbf{S}(t) = [1, 0, 1]$ *w.p.* $1 - \gamma$ *and*
  2) $\mathbf{S}(t) = [0, 1, 0]$ *w.p.* γ.
- *Else,* $\mathbf{S}(t) = \boldsymbol{\zeta}(t)$.

Clearly, this policy is a randomization between the two 3-queue MSM policies $\tilde{\pi}^{(3)}_{IQ}$ and $\pi^{(3)}_{OQ}$. A comparison of the definitions of $\rho^{(3)}_{\gamma}$, $\tilde{\pi}^{(3)}_{IQ}$ and $\pi^{(3)}_{OQ}$ clearly shows that $\rho^{(3)}_{\gamma}$ essentially chooses $\tilde{\pi}^{(3)}_{IQ}$ w.p. γ and $\pi^{(3)}_{OQ}$ w.p. $1 - \gamma$. By this we mean that in every time slot, $\tilde{\pi}^{(3)}_{IQ}$ and $\pi^{(3)}_{OQ}$ choose their actions and $\rho^{(3)}_{\gamma}$ then selects the former w.p. γ and the latter

[5]We use ρ instead of π to highlight the fact that this is a *randomized* policy.

w.p. $1-\gamma$. In [23, Sec. VI] we analyze this policy and establish two important results: (i) that $\rho_\gamma^{(3)}$ is unstable for $\gamma \in [0, 0.5)$. We also conjecture that $\rho_\gamma^{(3)}$ is unstable for all $\gamma \in [0, 1)$ but are unable to prove this as of now. Simulation results (Sec. IX) seem to indicate this as well, and (ii) Consider the set of arrival rates

$$\Lambda_\gamma^{(3)} := \left\{ \boldsymbol{\lambda} \in \mathbb{R}_+^3 \middle| \lambda_1 + \lambda_2 < \gamma, \lambda_2 + \lambda_3 < \gamma \right\}. \tag{12}$$

For every $\gamma \in (0, 1]$, $\rho_\gamma^{(3)}$ stabilizes all rate vectors in $\Lambda_\gamma^{(3)}$. It follows that the stability region of $\rho_\gamma^{(3)} \nearrow \Lambda_3$ as $\gamma \uparrow 1$.

V. Path-Graph Models with $N > 3$: Policy Splicing for Throughput Optimal QNB Scheduling

The previous section introduced scheduling policies that rely only on the empty-nonempty status of queues and examined the behavior of such policies on a small network. We will use the knowledge gained therein to now propose such policies for larger systems while still confining ourselves to path-graph interference networks. We introduce a "policy splicing" technique to construct MSM policies for large systems by splicing together MSM policies for smaller systems.

A. Top-down and Bottom-up scheduling on N queues

Recall the "Top-Down" and "Bottom-Up" policies, $\pi_{TD}^{(3)}$ and $\pi_{BU}^{(3)}$, discussed in Sec. IV.For a general path-graph network with N queues, the "Top-Down" policy, $\pi_{TD}^{(N)}$, which maps an occupancy vector $\boldsymbol{\zeta}(t)$ to an activation vector $\mathbf{s}(t)$, is defined as follows. Before defining the policy, we assume the presence of two *virtual queues*, Queue 0 and Queue $N+1$, with $Q_0(t) = Q_{N+1}(t) = s_0(t) = s_{N+1}(t) = 0$, $\forall t \geq 0$. This is just to facilitate compact writing of the policy. These virtual queues do not play any actual role in the system. Recall that if $\mathbf{Q}(t) = [Q_1(t), \cdots, Q_N(t)]$ is the queue length vector at time t, then the **occupancy vector** at time t is defined by $\boldsymbol{\zeta}(t) = [\mathbb{I}_{\{Q_1(t)>0\}}, \cdots, \mathbb{I}_{\{Q_N(t)>0\}}]$. The $\pi_{TD}^{(N)}$ policy: *At time t*

- *For j=1:N*
  1) *If* $\zeta_j(t) = 1$ *and* $s_{j-1}(t) = 0$, *then* $s_j(t) = 1$.
  2) *Else if* $\zeta_j(t) = 1$ *and* $s_{j-1}(t) = 1$, *then* $s_j(t) = 0$.
  3) *Else if* $\zeta_j(t) = 0$, *then* $s_j(t) = 0$.

It is easy to see that this produces $\pi_{TD}^{(3)}$ for $N = 3$, and $\pi_{BU}^{(N)}$ is defined similarly. The following important property follows from the definition.

Proposition 7. $\pi_{TD}^{(N)}$ and $\pi_{BU}^{(N)}$ are MSM for all $N \in \mathbb{N}$.

Proof. Refer Sec. XI-I in the supplementary material. ■

Before we venture into proving the throughput-optimality of $\pi_{TD}^{(N)}$ and $\pi_{BU}^{(N)}$, we use these two policies to describe the policy splicing process (see Fig. 3).

B. Splicing TD and BU policies

Consider a system of $2N-1$ queues, $N \geq 1$. In Algorithm 1, we splice the TD and BU policies and construct a scheduling policy[6] $\pi_{SP}^{(2N-1)}$ on this system. **Note:** We assume the presence of the two *virtual queues* Queue 0 and Queue $2N$ here as well.

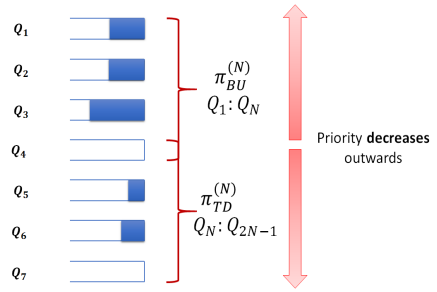


Fig. 3: Illustrating the manner in which the policies $\pi_{TD}^{(N)}$ and $\pi_{BU}^{(N)}$ are spliced together to form the *non-MSM* policy $\pi_{SP}^{(2N-1)}$. Note that the splicing is consistent in that even though the two sub-policies schedule over overlapping sections of queues, their decisions do not contradict each other.

Input: Binary occupancy vector $\boldsymbol{\zeta}(t)$;
Output: Queue activation vector $\mathbf{S}(t)$;
Initialize: $j = 1$, time$= t$, $\mathbf{S}(t) = \mathbf{0}$;
if $\zeta_N(t) = 1$ **then**
  $S_N(t) = 1$, $S_{N-1}(t) = 0$ and $S_{N+1}(t) = 0$
else
  for $j = N-1 : 1$ **do**
    if $\zeta_j(t) = 1$ *and* $S_{j+1}(t) = 0$ **then**
      $S_j(t) = 1$
    else
      if $\zeta_j(t) = 1$ *and* $S_{j+1}(t) = 1$ **then**
        $S_j(t) = 0$
      else
        if $\zeta_j(t) = 0$ **then**
          $S_j(t) = 0$
        end
      end
    end
  end
  for $j = N+1 : 2N-1$ **do**
    if $\zeta_j(t) = 1$ *and* $S_{j-1}(t) = 0$ **then**
      $S_j(t) = 1$
    else
      if $\zeta_j(t) = 1$ *and* $S_{j-1}(t) = 1$ **then**
        $S_j(t) = 0$
      else
        if $\zeta_j(t) = 0$ **then**
          $S_j(t) = 0$
        end
      end
    end
  end
end

Algorithm 1: The spliced policy $\pi_{SP}^{(2N-1)}$. The loop corresponding to $j = N-1 : 1$ induces $\pi_{BU}^{(N)}$ on Queues N through 1, while the latter loop induces $\pi_{TD}^{(N)}$ on Queues N through $2N$, as depicted in Fig. 3.

Before proceeding to analyze this policy, we first need to make sure it really is a well-defined policy, i.e., it provides a valid activation vector for each of the 2^{2N-1} possible occupancy vectors.

Lemma 8. $\pi_{SP}^{(2N-1)}$, as defined above, is well-defined.

Proof. See Sec. XI-K in the supplementary material. ■

A quick comparison with the definitions of $\pi_{TD}^{(N)}$ and $\pi_{BU}^{(N)}$ shows that $\pi_{SP}^{(2N-1)}$ induces the former two policies on the subsets $\{N, N+1, \cdots, 2N-1\}$ and $\{1, 2, \cdots, N\}$ respectively. The following result follows from the definition of the splicing process. Recall from Sec. II-B that the capacity region of

[6]The subscript "SP" refers to the fact that this is a *S*pliced *P*olicy.

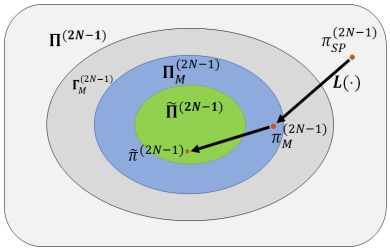


Fig. 4: Illustrating steps 3 and 4 of the general policy splicing process. The QNB, non-MSM policy $\pi_{SP}^{(2N-1)}$ is first projected into $\Pi_M^{(2N-1)}$ to get the *MSM* policy $\pi_M^{(2N-1)} \equiv L\left(\pi_{SP}^{(2N-1)}\right)$. Thereafter, $\pi_M^{(2N-1)}$ is modified to prioritize inner queues to get $\tilde{\pi}^{(2N-1)} \in \tilde{\Pi}^{(2N-1)}$.

a path-graph interference network consisting of N queues is defined by

$$\Lambda_N := \left\{ \lambda \in \mathbb{R}_+^N \middle| \lambda_i + \lambda_{i+1} < 1,\ 1 \le i \le N-1 \right\}, N \in \mathbb{N}. \quad (13)$$

Theorem 9. For every $N \in \mathbb{N}$, such that $\pi_{TD}^{(N)}$ and $\pi_{BU}^{(N)}$ are throughput-optimal over Λ_N, $\pi_{SP}^{(2N-1)}$ is throughput-optimal over Λ_{2N-1}.

PROOF. See Sec. XI-J in the supplementary material. ■

Remark. We will discuss the stability of our TD and BU policies in Sec. V-D. It is important to note that although $\pi_{TD}^{(N)}$ and $\pi_{BU}^{(N)}$ are MSM, $\pi_{SP}^{(2N-1)}$ is *not*. For example, consider an occupancy vector such that $\zeta_{N-1}(t) = \zeta_N(t) = \zeta_{N+1}(t) = 1$ and $\zeta_j(t) = 0,\ \forall j \in \{1, \cdots, N-2\} \cup \{N+2, \cdots, 2N-1\}$, i.e., the central queue and both adjacent queues are nonempty and all other queues are empty. Any MSM policy would produce the activation vector with $S_{N-1}(t) = S_{N+1}(t) = 1$ and $S_j(t) = 0,\ \forall j \notin \{N-1, N+1\}$, whereas $\pi_{SP}^{(2N-1)}$ produces the activation vector with $S_N(t) = 1$ and $S_j(t) = 0,\ \forall j \neq N$ thereby scheduling one less queue for transmission.

C. Mapping $\pi_{SP}^{(N)}$ to an MSM policy

To reduce delay, one needs to extract an MSM policy from this spliced policy. Informally, we project the policy onto the space $\Pi_M^{(N)}$ of MSM policies using the projection operator $L(\cdot)$, described in Sec. IV-A and defined in [23, Sec. V.C]. Thereafter, we use some observations based on Condition 2 in Sec. III-A to improve the delay performance of this projected MSM policy to finally obtain a policy in $\tilde{\Pi}^{(N)}$. Figures 3 and 4 give a pictorial description of the entire process.

Remark. Another important observation is that $\pi_{SP}^{(2N-1)}$ is also a stabilizing policy for a system with $2N-k$ queues, with $1 \le k \le 2N$. One simply needs to begin with a system of $2N-k$ queues and append k virtual queues that start out empty and receive no arrivals in any time slot and run $\pi_{SP}^{(2N-1)}$ on them. So, the focus of the remainder of this section is proving the throughput-optimality of the Top-Down and Bottom-Up priority policies.

To summarise, the general policy splicing process involves the following steps (see Figures 3 and 4).

General splicing procedure:

1) Proposing Top-Down (TD) and Bottom-Up (BU) policies for the N-queue system,
2) Splicing them together to produce a non-MSM policy for the $(2N-1)$-queue system,
3) Projecting the spliced policy to get an MSM policy,
4) Modifying this policy to break ties in favor of inner queues to get a policy with better delay performance than its MSM predecessor. For example, with $N = 5$ queues and $\boldsymbol{\zeta}(t) = [1,1,1,1,0]$, policies that choose $\mathbf{S}(t) = [1,0,1,0,0]$ and $\mathbf{S}(t) = [0,1,0,1,0]$ are both MSM, but only the latter breaks ties in favor of inner queues[7].

We have already proposed and analyzed two priority policies for 3-queue path graph networks that we named $\pi_{TD}^{(3)}$ and $\pi_{BU}^{(3)}$. As a quick illustration of the above procedure, in [23, Sec. VI.A] we explicitly show the entire procedure of how low delay policies for $2 \times 3 - 1 = 5$-queue systems can be constructed from these two 3-queue policies. Due to space constraints, we are unable to present the details in this paper.

D. Policies for $N = 7,\ 8$ and 9 through Policy Splicing

The Top-Down and Bottom-Up policies $\pi_{TD}^{(4)}$ and $\pi_{BU}^{(4)}$ will, as already discussed, be used to develop stabilizing policies for systems with $N = 2\times4-1 = 7$ queues. An equivalent way to define the Top-Down policy $\pi_{TD}^{(4)}$ is as below.

At time t:

1) *If* $Q_1(t) > 0$,
   a) *If* $Q_3(t) > 0$, $\mathbf{s}(t) = (1,0,1,0)$.
   b) *Else*[8], $\mathbf{s}(t) = (1,0,0,1)$.
2) *Else, if* $Q_2(t) > 0$, $\mathbf{S}(t) = (0,1,0,1)$.
3) *Else*,
   a) *If* $Q_3(t) > 0$ $\mathbf{S}(t) = (1,0,1,0)$.
   b) *Else*, $\mathbf{S}(t) = (1,0,0,1)$.

Proposition 10. $\pi_{TD}^{(4)}$ is throughput optimal.

PROOF. Notice that as far as the subsystem $[Q_1(t), Q_2(t), Q_3(t)]$ is concerned, this policy reduces to $\pi_{TD}^{(3)}$. That is, $\pi_{TD}^{(4)}$ restricted to the first three queues is $\pi_{TD}^{(3)}$. So, that subsystem is strongly stable.

The remainder of the proof of the proposition can be found in Sec. XI-L in the supplementary material. ■

It is easy to see that the Bottom-Up policy, $\pi_{BU}^{(4)}$, defined in a symmetric manner, giving highest priority to Queue 4 and lowest to Queue 1, is also T.O. We then define $\pi_{SP}^{(7)}$ as described in Sec. 1. Clearly, since $\pi_{SP}^{(7)}$ restricted to Queues 1, 2, 3, 4 is just $\pi_{BU}^{(4)}$ and restricted to Queues 4, 5, 6, 7 is $\pi_{TD}^{(4)}$, using the fact that both $\pi_{TD}^{(4)}$ and $\pi_{BU}^{(4)}$ are throughput-optimal and Thm. 9, we conclude that $\pi_{SP}^{(7)}$ is throughput-optimal as well. However, since $\pi_{SP}^{(7)}$ is not MSM, some modifications are required to improve delay performance. This simply requires executing Steps 3 and 4 in the general splicing procedure V-C.

[7]For further details, see Lem. 19 in the Appendix

[8]Strictly speaking, from the definition of $\pi_{TD}^{(N)}$ above, $s_4(t)$ should be set to 1 iff $\zeta_4(t) = 1$. But setting $s_4(t) = 1$ when $\zeta_4(t) = 0$ doesnt violate interference constraints since it means that Queue 4 is empty.

In a similar manner we will now show that the top-down and bottom-up policies for the 5 queue system ($\pi^{(5)}_{TD}$ and $\pi^{(5)}_{BU}$) are both throughput-optimal, which will immediately yield a stabilizing policy ($\pi^{(9)}_{SP}$) for the 9-queue system.

Proposition 11. $\pi^{(5)}_{TD}$ is throughput-optimal.

PROOF. This analysis closely follows our analysis of $\pi^{(4)}_{TD}$. With $\pi^{(5)}_{TD}$ we only need to prove that Queue 5 receives "enough" service, since this policy restricted to the first 4 queues is just $\pi^{(4)}_{TD}$ which, as we have just shown, is throughput-optimal. The remainder of the proof of the proposition can be found in Sec. XI-N. ■

The analysis of the bottom-up policy ($\pi^{(5)}_{BU}$) proceeds in a symmetric fashion. This means that the 9-queue policy $\pi^{(9)}_{SP}$, that induces $\pi^{(5)}_{TD}$ on queues 1, 2, 3, 4, and 5 and $\pi^{(5)}_{TD}$ on queues 5, 6, 7, 8 and 9, is throughput-optimal.

To summarize, in this section, we first proposed a general procedure to generate MSM Top-Down and Bottom-Up priority policies for a system with any $N \in \mathbb{N}$ queues. We then showed how these policies can be combined to construct policies for larger systems and provided a sufficient condition for such a spliced policy to be throughput-optimal. We then spliced the TD and BU policies for $N = 4$ and $N = 5$ queue systems and constructed *stable* QNB-MSM policies for systems with up to $N = 9$ queues. We now move on to analyzing the queueing delay performance of these policies.

VI. Path-Graph Conflict Graphs with $N > 3$: Delay with QNB Policies

We now turn our attention to the vital aspect of delay. We have already proved, in Thm. 4, that for the system with $N = 3$ queues, there exists a (unique) uniformly delay-optimal policy, that we named $\tilde{\pi}^{(3)}_{IQ}$. The natural question to ask in this context is if one can find a delay optimal queue nonemptiness-based policy for larger systems as well. In this section, we will answer this question in the *negative.*

Theorem 12. For all $N \geq 4$, there does not exist any policy in $\Pi^{(N)}_M$ that is uniformly delay optimal over all of Λ^o_N.

The proof of this theorem has been omitted due to space constraints, but we provide a short sketch below. The main idea is to show that $\Pi^{(4)}$ does not contain any queue length agnostic policy that is uniformly delay optimal over the entire throughput capacity region Λ_4 (the complete procedure is detailed in [23, Sec. IX]). We begin by proving, in Prop. 19 in [23], that the class of inner-queue prioritising MSM policies, $\tilde{\Pi}^{(4)}$, does not contain any uniformly delay optimal policy. $\tilde{\Pi}^{(4)}$ contains four policies which we show to be throughput optimal by first showing them to be the result of a splicing procedure involving one of the two priority policies $\pi^{(3)}_{TD}$ or $\pi^{(3)}_{BU}$, and $\pi^{(3)}_{IQ}$ (defined in Sec. IV-C). We then prove a theorem similar to Thm. 9 to establish throughput optimality. However, we then show that none of these policies performs uniformly better (or at least as good as) the others over all of Λ_4.

Next, in Prop. 20 therein we show that policies in $\tilde{\Pi}^{(4)}$ show better delay performance than those in $\Pi^{(4)}_M$. We already know from (9) that the delay of any policy in $\Pi^{(N)}$ can be improved by projecting it onto $\Pi^{(N)}_M$. Thus, Prop. 20 in [23], along with (9), shows that delay optimal policies, when they exist, must necessarily lie in $\tilde{\Pi}^{(N)}$. This observation, along with the nonexistence of delay optimal policies in $\tilde{\Pi}^{(4)}$ are used to prove the claim in Thm. 12.

VII. Cluster-of-Cliques Interference Networks: Throughput Optimal Scheduling

We will now show that some of the scheduling policies developed for path-interference graph networks extend in a natural manner to policies for the SoC and the LAoC networks. **Notation:** We denote policies designed for Star-of-Cliques networks by "ϕ" and include an "(S)" in the superscript to emphasize this. On the other hand, "θ" with and "(L)" in the superscript specifies an LAoC network policy. We will begin with centralized scheduling in SoC networks.

A. Scheduling in Star-of-Cliques Networks

Consider the following policy that we denote $\tilde{\phi}^{(S)}_{IC}$, which is motivated by the 3-node path graph policy $\tilde{\pi}^{(3)}_{IQ}$ which we discussed in Sec. IV-A. Recall that we defined N to be the total number of queues in the network. In keeping with the objective of developing queue nonemptiness-based policies, in every slot, $\tilde{\phi}^{(S)}_{IC}$ maps the occupancy vector $\boldsymbol{\zeta}(t) \in \{0,1\}^N$ to an activation vector $\mathbf{s}(t) \in \{0,1\}^N$. We define $\tilde{\phi}^{(S)}_{IC}$ as follows.

At each time t :

1) *If $\prod_{m=2}^{N} \left(\sum_{l \in C_m} \zeta_l(t)\right) > 0$ serve any nonempty queue in every clique $\{C_m,\ m \geq 2\}$ having nonempty queues.*
2) *Else, if $\sum_{l \in C_1} \zeta_l(t) > 0$, serve any nonempty queue in C_1.*
3) *Else, serve one nonempty queue (if it exists) in each of $\{C_m,\ m \geq 2\}$.*

In words, the above policy states that at time t,

- if every peripheral clique has at least one non empty queue, then serve one non empty queue in each of these cliques,
- else, if the inner clique has a non empty queue, serve one non empty queue in that clique,
- else, serve one non empty queue in every peripheral clique that has a non empty queue.

Proposition 13. $\tilde{\phi}^{(S)}_{IC}$ is throughput-optimal.

PROOF. The main idea behind the proof of this proposition is to prove a more general version of Property $\mathcal{P}$ (which we defined in Lem. 1 for path-graph networks) and use and use the total per-clique backlog as inputs to a new Lyapunov function to prove strong stability. See XI-O for details. ■

Towards the end of our discussion on queue nonemptiness-based scheduling for path-graph networks with $N = 3$ queues (see Sec. IV-C), we defined a *non-MSM* policy $\pi^{(3)}_{IQ}$. Extending this to the SoC network model gives us a second queue nonemptiness-based policy $\phi^{(S)}_{IC}$, which we define as follows.

At time t,

1) *If $\sum_{l \in C_1} \zeta_l(t) > 0$, serve any nonempty queue in C_1.*
2) *Else, serve one nonempty queue (if it exists) in each of $\{C_m,\ m \geq 2\}$.*

Proposition 14. $\phi_{IC}^{(S)}$ is throughput-optimal.

PROOF. Once again, the proof of this result rests on proving the new version of $(\mathcal{P})$ for this policy, followed by Lyapunov analysis. The proof is available in Sec. XI-T. ■

Remark. We end this section with some remarks about implementation and delay performance. From the point of view of implementation, the latter, $\phi_{IC}^{(S)}$ is actually easier to implement than $\tilde{\phi}_{IC}^{(S)}$. We discuss this in detail in Sec. VIII which is completely dedicated to implementation issues. However, $\tilde{\phi}_{IC}^{(S)}$ has its own advantages. With respect to the packet delay, recall that we had used a stochastic ordering argument to prove the delay optimality of Policy $\tilde{\pi}_{IQ}^{(3)}$ and later used a similar technique to show the absence of uniformly delay-optimal queue nonemptiness-based policies for path-graph networks. Along similar lines, we compare the delays induced by $\tilde{\phi}_{IC}^{(S)}$ and $\phi_{IC}^{(S)}$ below.

1) Comparison of delay with $\tilde{\phi}_{IC}^{(S)}$ and $\phi_{IC}^{(S)}$:

Proposition 15 (Stochastic dominance of system backlog)**.** Let the system backlog at time $t \geq 0$ with $\phi_{IC}^{(S)}$ and $\tilde{\phi}_{IC}^{(S)}$ be denoted by $\mathbf{Q}^{\phi_{IC}^{(S)}}(t)$, and $\mathbf{Q}^{\tilde{\phi}_{IC}^{(S)}}(t)$ respectively. Then, with $\mathbf{Q}^{\tilde{\phi}_{IC}^{(S)}}(0) \stackrel{s}{=} \mathbf{Q}^{\phi_{IC}^{(S)}}(0)$, and arrivals to corresponding queues having the same statistics in both systems,

$$\sum_{m=1}^{N}\sum_{j=1}^{N_m} Q_{m,j}^{\tilde{\phi}_{IC}^{(S)}}(t) \stackrel{st}{\leq} \sum_{m=1}^{N}\sum_{j=1}^{N_m} Q_{m,j}^{\phi_{IC}^{(S)}}(t), \ \forall t \geq 0. \tag{14}$$

PROOF SKETCH. This proof proceeds along the same lines as the proof of delay optimality of Policy $\tilde{\pi}^{(3)}$ that we presented in Sec. XI-G. It can be found in Sec. XI-R in the supplementary material. ■

We now begin our study of scheduling in LAoC networks, and return to policies for Star-of-Cliques networks once again when we shift our focus to decentralized implementation.

B. Scheduling in Linear-Arrays-of-Cliques

The technique we use to propose scheduling policies for LAoC networks is the policy splicing technique we developed in Sec. V. The proofs therein cannot be directly used to assess the stability of policies designed for LAoC networks since the proofs are designed for Bernoulli arrival processes to queues and require some more work to be extended to handle scheduling over cliques: however, one could argue that a clique can, in essence, be treated as a queue with an arrival process which is simply the sum of the arrivals to the constituent queues. For example, Clique C_1 in Fig. 2a can be treated as a single queue with an arrival process that is the sum of the processes to Queues $Q_{1,1}, Q_{1,2}$ and $Q_{1,3}$ therein. The resulting arrival process to the queue would then be a *batch* arrival process with *arbitrary batch size* (there can be any number of queues in a clique), and simple extensions of the proofs supplied hitherto can be shown to suffice.

As before, we begin with Top-Down and Bottom-Up policies for the 3-clique LAoC and splice them to construct policies for the LAoC's with 4 and 5 cliques. Note, once again, that we place no restrictions on the number of queues within any clique.

1) Scheduling Policies for Systems with $N = 3$ Cliques: The policy $\theta_{TD}^{(3L)}$ is described as follows.

At time t :

- *If $\sum_{j=1}^{N_1} \zeta_{1,j}(t) > 0$ schedule any non-empty queue in C_1.*
  - *If $\sum_{j=1}^{N_3} \zeta_{3,j}(t) > 0$ schedule any non-empty queue in C_3.*
- *Else, if $\sum_{j=1}^{N_2} \zeta_{2,j}(t) > 0$ schedule any non-empty queue in C_2.*
- *Else schedule any non-empty queue in C_3.*

In other words, if in slot t

- there is a non-empty queue in C_1, then $\theta_{TD}^{(3L)}$ serves one non-empty queue in C_1 and C_3.
- if C_1 is empty but C_2 has a non-empty queue in it, then $\theta_{TD}^{(3L)}$ serves that queue.
- if C_1 and C_1 are both empty, then $\theta_{TD}^{(3L)}$ serves any non-empty queue in C_3.

Proposition 16. $\theta_{TD}^{(3L)}$ is throughput-optimal.

PROOF. This proof uses the ideas involved in proving the throughput-optimality of $\pi_{TD}^{(3)}$ and simply extends them to incorporate batch arrivals. The proof is available in Sec. XI-S. ■

A similar proof shows that $\theta_{BU}^{(3L)}$ is also throughput-optimal.

2) Scheduling Policies for Systems with $N = 4$ and 5 Cliques: Now, by splicing together $\theta_{TD}^{(3L)}$ and $\theta_{BU}^{(3L)}$, one can construct stable policies for the system with 5 cliques and hence, systems with 4 cliques. The spliced policy, $\theta_{SP}^{(5L)}$ is defined as

At time t :

1) *If $\sum_{j=1}^{N_3} \zeta_{3,j}(t) > 0$ then schedule a nonempty queue in C_3.*
   a) *If $\sum_{j=1}^{N_1} \zeta_{1,j}(t) + \sum_{j=1}^{N_5} \zeta_{5,j}(t) > 0$ then schedule any nonempty queue each in C_1 and C_5.*
2) *Else if $\sum_{j=1}^{N_2} \zeta_{2,j}(t) \times \sum_{j=1}^{N_4} \zeta_{4,j}(t) > 0$ then schedule a nonempty queue each in C_2 and C_4.*
3) *Else if $\sum_{j=1}^{N_2} \zeta_{2,j}(t) \times \sum_{j=1}^{N_5} \zeta_{5,j}(t) > 0$ then schedule a nonempty queue each in C_2 and C_5.*
4) *Else if $\sum_{j=1}^{N_4} \zeta_{4,j}(t) \times \sum_{j=1}^{N_1} \zeta_{1,j}(t) > 0$ then schedule a nonempty queue each in C_1 and C_4.*
5) *Else schedule any nonempty queue each in C_1 and C_5.*

Proposition 17. The policy $\theta_{SP}^{(5L)}$ is throughput-optimal.

PROOF. See Sec. XI-P. ■

To summarize, in this section, we studied scheduling in the Star-of-Cliques and Linear-Array-of-Cliques models that could occur in IoT-type sensor network applications. Having characterized the capacity region of such networks, we proposed and analyzed multiple scheduling policies. However, as mentioned before, these policies depend on being able to find a nonempty queue in every slot in which the system is not empty. While disseminating occupancy information across the network is certainly not as expensive as sharing queue length information (required by, say, the MaxWeight family of scheduling algorithms), it raises the question of the existence of policies can work with even less information. The following section describes some preliminary attempts at achieving this goal.

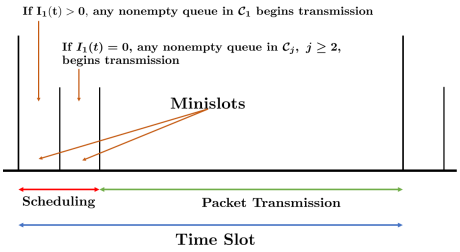


Fig. 5: Implementing $\phi_{IC}^{(S)}$ using the slot and minislot structures. It is important to note that the number of minislots is O(1), i.e., *does not scale* with the number of communication links the network. This results in constant scheduling overhead, which is patently desirable.

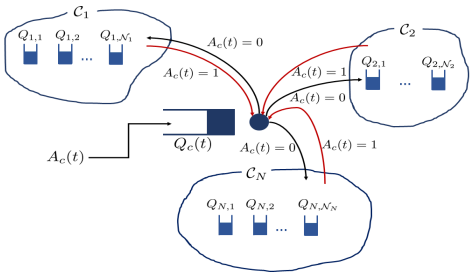


Fig. 6: The system upon which the policy $\phi_{CS}^{(S)}$ is analyzed. $Q_c(t)$ is the central queue which has highest priority. This means that the queue is served whenever it has packets, i.e., whenever the arrival $A_c(t) = 1$. This is indicated by the **red arrows** that show access to the channel being granted by $\phi_{CS}^{(S)}$ to Q_c. However, when $Q_c(t) = 0$, $\phi_{CS}^{(S)}$ enters Step 2 and serves the appropriate queue in the peripheral cliques, as indicated by the **black arrows**.

VIII. Some Remarks on Decentralized Implementation

In this section, we discuss several ways in which the policies developed and analyzed hitherto can be made amenable to decentralized implementation. QNB policies, by definition, only require the empty-non empty statuses of queues. While this vector, $\zeta(t)$ by itself can be disseminated across the network using N bits[9] in many cases this status can be *inferred* using a simple hypothesis test and requires no information exchange between queues. To accomplish this decentralized state inference, we take the help of what are known as *minislots* [16], [33] which we describe in detail, below. Note that while we focus on SoC networks in this section, we also address decentralized scheduling in LAoC networks in [23, Sec. XI.D].

Transmission Sensing: We assume that all nodes transmit at the same fixed power, and that the maximum internode distance is such that every node in clique $C_j, j \geq 2$ can sense the power from a transmitting node in (the central) clique C_1 and vice versa, as dictated by the interference constraints[10]. Suppose a node has been scheduled to transmit in a slot. Then, whether or not the node actually transmits can be determined by the other nodes by averaging the received power over a small time interval (akin to the "Clear Channel Assessment" or CCA mechanism [34]). For reliable assessment, the interval will need to be of a certain length, and the distance between the nodes will need to be limited. As before, we refer to this activity-sensing interval a **minislot** (see [16], [33] and Fig. 5). Decentralized methods of implementing both $\phi_{IC}^{(S)}$ and $\tilde{\phi}_{IC}^{(S)}$ follow from the minislot structure. Define the *occupancy* of Queue k in Clique j by $i_{j,k}(t) := \mathbb{I}_{\{Q_{j,k}(t)>0\}}$, and let $I_j(t) := \sum_{k \in C_j} i_{j,k}(t)$, $j \geq 1$, indicate if Clique j has any nonempty nodes at the beginning of time slot t. We will discuss implementing $\phi_{IC}^{(S)}$ in detail, and do the same for $\tilde{\phi}_{IC}^{(S)}$ in [23, Sec. XI.C].

A. Decentralized Implementation of $\phi_{IC}^{(S)}$

At time t,

1) *If $I_1(t) > 0$, then one nonempty node from clique C_1 is allowed to transmit (see Fig. 5). Nodes in the other cliques sense this transmission in the first minislot and refrain from transmitting during that slot.*
2) *If no power is sensed in the first minislot, it means $I_1(t) = 0$, and each of the other cliques choose one nonempty queue (if any) for transmission during that slot.*

This, of course, assumes that one is somehow able to identify a nonempty queue, if one exists, in each clique. So this implementation is, by itself, centralized *within* a clique and *decentralized* across cliques. We now propose methods to determine which (nonempty) queue *within* a clique actually gets to transmit in either of the two steps above.

One method is for the nodes in a clique to periodically share occupancy information which could be accomplished by having a sink node in every clique. The sink node of each clique periodically aggregates occupancy information from its nodes and uses it to schedule nonempty queues in some order. We discuss this towards the end of this section, and in Sec. VIII-B we propose and analyze a version of $\phi_{IC}^{(S)}$ that requires no explicit information exchange between queues.

B. $\phi_{IC}^{(S)}$ without Occupancy Information: Towards Fully Decentralized Policies

First consider a clique, say C_i, in isolation. This is, by itself, a fully connected interference graph. Suppose the nodes in C_i could determine the backlog of a node in C_i each time it transmitted a packet[11]. Then, at the beginning of slot t, the information common to all nodes in C_i would consist of the number of slots $V_i(t)$ since node i last transmitted[12] and its backlog $Q_i(t - V_i(t))$ at that instant. With this partial information structure and equal arrival packet rates to the queues, we have already shown in [16], that exhaustively serving a nonempty queue minimizes mean delay. With exhaustive service, $Q_i(t - V_i(t))$ is always 0, which obviates the need to transmit queue lengths. When the queue under service, called the **incumbent** in the sequel, becomes empty we have already shown that scheduling node $arg\ max_{j \in C_i} V_j(t)$ is throughput-optimal, and under certain conditions, also mean delay optimal. Motivated by this, we define another partial

[9]This could potentially even be reduced to $\log_2(N)$ bits using network coding techniques – this is an important direction for future work.

[10]Obviously, for all $2 \leq j, k \leq N$, nodes in C_j cannot sense transmissions in C_k and vice versa.

[11]The backlog information could be quantized and contained in the packet header, for example.

[12]If the node were empty at this instant, it wouldn't have actually transmitted anything. The others can infer its "emptiness" by sensing no power in a minislot.

information version $\phi^{(S)}_{CS}$, of $\phi^{(S)}_{IC}$ below **under the assumption** that the inner clique C_1 has exactly *one* node, i.e., $|C_1| = 1$. We refer to this queue as Queue c (see Fig. 6).

At time t :

1) *If $I_1(t) > 0$, then the queue in C_1 transmits its packet. Nodes in the other cliques sense this transmission in minislot 1 (see Fig. 5) and refrain from attempting any transmissions.*
2) *If no power is sensed in minislot 1, every clique C_i, $i \geq 2$ does the following*
   a) *The incumbent begins to transmit at the end of minislot 1 if it is nonempty. The other nodes in C_i sense this in minislot 2 and refrain from attempting transmissions.*
   b) *If no power is sensed in minislot 2, then the incumbent is empty and $arg\ max_{j \in C_i} V_j(t)$ is now allowed to transmit.*

Adding more minislots reduces chances of slot wastage, but also reduces system throughput since it increases time wasted in not actually transmitting a packet. Hence, this parameter represents a tradeoff between throughput and delay. It is not clear if this policy is throughput-optimal and we now provide a formal argument.

Theorem 18. The policy $\phi^{(S)}_{CS}$ is throughput-optimal.

PROOF SKETCH. The proof uses a Lyapunov drift argument and invokes the Foster-Lyapunov theorem to prove that the system backlog process $\mathbf{Q}(t) := [Q_1(t), \cdots, Q_N(t)]$, $t \geq 0$ is positive recurrent. The details of the proof can be found in the supplementary material in Sec. XI-U. ■

A couple of remarks before we present numerical results.

Remark. In [23, Sec. XI.D] we discuss how to implement $\tilde{\phi}^{(S)}_{IC}$ in a decentralized manner. As mentioned before, another commonly used technique to reduce control traffic for scheduling is to disseminate state information *periodically*. In [23, Sec. XI.A,D] we develop a family $\left\{\Phi^{(S)}_{IC}(T),\ T \geq 1\right\}$ of such policies that take scheduling decisions based control information obtained only every T time slots. We show that the family is throughput-optimal and also that they display some interesting delay properties with respect to the dissemination period, T.

IX. SIMULATION RESULTS

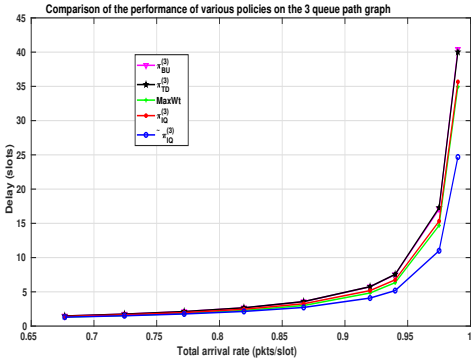
In this section we numerically compare the performance of the various policies we have proposed and analyzed in the preceding sections. To begin with, we simulate the mean delay performances of the policies for the path graph network with $N = 3$, discussed in Sec. IV and compare them against the MaxWeight scheduling policy. To recapitulate, $\pi^{(3)}_{TD}$, and $\pi^{(3)}_{BU}$ are the Top-Down and Bottom-Up policies respectively, $\tilde{\pi}^{(3)}_{IQ}$ is the delay optimal policy defined in IV-A and $\pi^{(3)}_{IQ}$ is the throughput-optimal *non-MSM* policy defined in IV-C. In every slot $t \geq 0$, MaxWeight simply serves Queues 1 and 3 if $Q_1(t) + Q_3(t) > Q_2(t)$ and Queue 2 otherwise. Obviously, this policy requires more state-information than any of the others. We simulate these policies when the arrival processes to the three queues are independent Bernoulli processes of rates $s \times [0.25, 0.74, 0.25]$, $s \in [0, 1)$, i.e., the inner queue has a high arrival rate, and $s \times [0.74, 0.25, 0.74]$, i.e., the outer two queues have the high arrival rates. The results are shown in Figures 7a and 7b. As claimed in Thm. 4, $\tilde{\pi}^{(3)}_{IQ}$ performs best, showing in mean delay of up to 30% less than MaxWeight near $s = 1$ (in fact, the reduction in delay becomes more pronounced as s approaches 1) and 38% less than $\pi^{(3)}_{TD}$. Notice that in both plots MaxWeight does not perform as well as $\tilde{\pi}^{(3)}_{IQ}$ showing that it does not prioritize the middle queue frequently "enough."

Moving on, although we have proved our stability results with Bernoulli arrival processes, we now provide a simulation study which suggests that these results seem to hold for more general arrival processes. Also note that the stochastic ordering proof of Thm. 4, i.e., the delay optimality of policy $\tilde{\pi}^{(3)}_{IQ}$, being a sample path optimality argument, does not take into account the fact that the arrival processes to the queues are Bernoulli. It is, hence, equally valid for other types of arrival processes as well. Our simulations bear out this fact. Once again, we simulate our policies on the 3-queue path-interference graph with Markovian arrival processes as described below. The arrivals to every queue form a two-state stationary discrete-time Markov chain (DTMC), i.e., $\{A_i(t),\ t \geq 1\}$ forms a DTMC. As before, $A_i(t) = 1$ refers to the arrival of a packet into Queue i and $A_i(t) = 0$ refers to no arrivals. Fig. 8 shows the transition probability diagram of a generic two-state DTMC. For Queue j, the stationary probability of the arrival being in State i, $i \in \{0, 1\}$ is given by $\xi_{i,j}$; obviously, $\xi_{0,j} + \xi_{1,j} = 1$, $\forall j \in \{1, 2, 3\}$. Suppose the transition probabilities of the process for Queue j are given by $P\{A_j(t+1) = 1 | A_j(t) = 0\} = p_j$, and $P\{A_j(t+1) = 0 | A_j(t) = 1\} = q_j$. From basic Markov chain theory, we know that $\xi_{1,j} = \frac{p_j}{p_j + q_j}$, $\xi_{0,j} = \frac{q_j}{p_j + q_j}$ and for every $t \geq 1$, $\mathbb{E}A_j(t) = \xi_{1,j} = \lambda_j$.

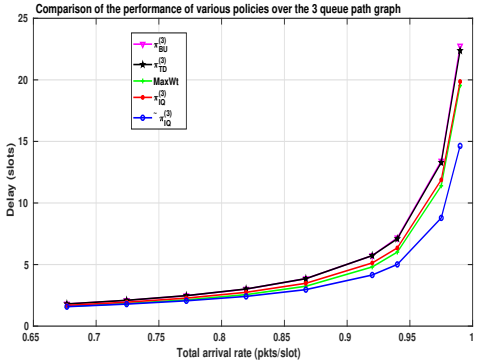
Both plots (Figures 9a and 9b) bear out the fact that even with non-Bernoulli arrivals, $\tilde{\pi}^{(3)}_{IQ}$ shows the best delay performance beating the closest competitor by at least 34%. Again, in both plots we see that MaxWeight performs about just as well as the Top-Down and Bottom-Up policies, suggesting that it does not prioritize the inner queue "enough." We now move on to the performance of the randomized policy $\rho^{(3)}_{\gamma}$, indexed by the randomization parameter $\gamma \in [0, 1]$. In Sec. IV-D we derived an inner bound on the set of arrival rates that the policy can stabilize for a given γ, its stability region[13], and showed that $\Lambda^{(3)}_{\gamma} \nearrow \Lambda_3$ as $\gamma \uparrow 1$. The plot in Fig. 10, simulated with $\gamma = 0.5,\ 0.55$, and 0.6 help corroborate our analysis. However, the plots also suggest that the inner bound $\Lambda^{(3)}_{\gamma}$ is actually *not very tight*. Further study is required to establish better bounds on this region.

Moving on to larger path graphs, recall that in Sec. V-C we proposed a policy-splicing procedure to derive low delay QNB-MSM scheduling policies for path graphs with arbitrary number of queues. We demonstrate the performance of these policies in Table II, where we compare our proposed policies

[13] See Sec. II-A for details.



(a) Delay performance of the policies $\tilde{\pi}^{(3)}_{IQ}$, $\pi^{(3)}_{IQ}$, MaxWeight, $\pi^{(3)}_{TD}$ and $\pi^{(3)}_{BU}$ along the trajectory $\boldsymbol{\lambda}(s) = s \times [0.25, 0.74, 0.25]$, $s \in [0, 1]$, in the capacity region Λ_3.



(b) Delay performance of the policies $\tilde{\pi}^{(3)}_{IQ}$, $\pi^{(3)}_{IQ}$, MaxWeight, $\pi^{(3)}_{TD}$ and $\pi^{(3)}_{BU}$ along the trajectory $\boldsymbol{\lambda}(s) = s \times [0.74, 0.25, 0.74]$, $s \in [0, 1]$, in the capacity region Λ_3.

Fig. 7: Simulation results for the path-graph network with $N = 3$ for *Bernoulli* packet arrival processes. The mean delay performances of all deterministic policies discussed in Sec. IV are shown in Figures (a) and (b), and compared with the MaxWeight scheduling policy [11].

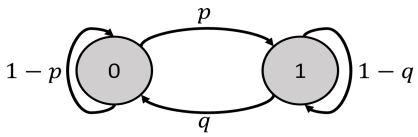


Fig. 8: The transition probability diagram of the Markovian arrival process. If, in slot t, the arrival process was in State 0, i.e., $A(t) = 0$, then in slot $t+1$, $A(t+1) = 1$ with probability p and $A(t+1) = 0$ with probability $1 - p$.

Number of queues (N)	Mean sum queue length (packets) Bernoulli arrivals		
	QNB-MSM $\left(\tilde{\pi}^{(N)}\right)$	MaxWeight	$L(MW\alpha)$
4	45.963	57.302	**43.508**
5	**61.537**	88.243	75.642
15	**76.72**	107.88	92.100

TABLE II: Path graph interference models with $N = 4, 5$ and 15. Comparison of sum queue length with Bernoulli arrivals, under the proposed QNB MSM policies with MaxWeight and $L(MW\alpha)$ with $\alpha = 0.01$. Details about the arrival rate vectors can be found in Sec. XI-V of the Appendix.

with the benchmark MaxWeight (MW) and a third policy that is based on a popular scheduler called the "MaxWeight-α" scheduler. This last policy, that we denote by $L(MW\alpha)$, is an MSM policy, obtained by using the operator L (see Sec. IV-A) to project a modification of MaxWeight (MW) called $MW\alpha$ onto $\Gamma^{(N)}_M$. The $MW\alpha$ policy, studied in [35] and [17], is essentially MW with all queue lengths raised to their α^{th} powers, with $\alpha > 0$. This policy has been observed to show smaller sum queue lengths (than MW) with smaller α [36].

The table shows that our proposed policies do outperform MaxWeight in all cases[14]. Note that the arrival rate vectors have not been shown in Table II due to space constraints. We have reported the vectors in Sec. XI-V of the Appendix. Recall that the analysis of throughput optimality was limited to $N = 9$ queue systems. In Row 3, we perform the splicing procedure (Sec. V-C) to produce a QNB-MSM policy for a system with $N = 15$ queues and show that it outperforms both the benchmark policies. Finally, Row 1 of the table shows an arrival rate vector for which our proposed queue length-agnostic policy does worse and $L(MW\alpha)$ shows the smallest sum queue length. In light of Thm. 12, this should not be entirely surprising. Moreover, the loss in performance is small.

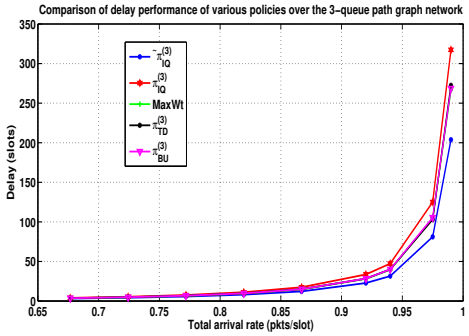
We move on to simulations of the policies proposed for the second class of conflict graphs discussed in this article, namely, Cluster-of-cliques graphs. The first, shown in Fig. 11, is a *Star-of-Cliques* (SoC) networks comprising 4 cliques and a total of 6 queues. The second network is the LAoC network shown in Fig. 2a. It consists of 4 cliques and a total of 9 queues. Table III shows the result of simulating $\tilde{\phi}$, $\tilde{\theta}^{(5L)}$ (the projected version of $\theta^{(5L)}_{SP}$, defined in Sec. VII-B) and MW on these networks. We see that the proposed policies consistently perform better than the benchmarks.

The result for the SoC networks is, in particular, quite interesting, since one expects that situations may arise wherein only two of the three peripheral cliques and C_1 are nonempty. In such a case, $\tilde{\phi}$ would serve C_1, giving up the chance to serve both the peripheral nonempty cliques simultaneously and remove 2 packets from the system in a single slot, which is what MW might have attempted, if the queues therein were large enough. If, for example, in some slot t, C_2 is empty, while $Q_{1,1}(t) = 1$, $Q_{3,1}(t) = 5$ and $Q_{4,1}(t) = 2$, $\tilde{\phi}$ still serves only $Q_{1,1}$ (1 packet transmitted) while MaxWeight serves both $Q_{3,1}$ and $Q_{4,1}$ (2 packets transmitted). Why $\tilde{\phi}$ still performs better requires more investigation and will be a focus of our future work.

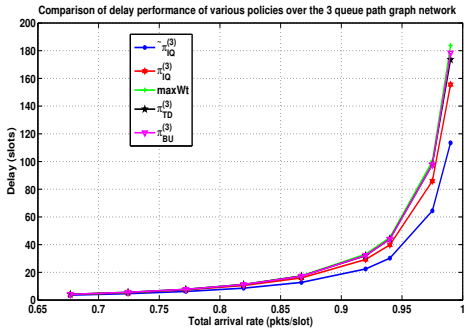
X. Conclusion and Future Work

In this paper, we began by studying the scheduling of transmissions over a class of noncollocated interference networks that we called "path-graph interference networks." We provided sufficient conditions for queue nonemptiness based (QNB) policies to be throughput-optimal over these networks. We then provided a complete characterization of the class of

[14]The construction of the QNB-MSM policy in the first row which, by our nomenclature, is denoted $\tilde{\pi}^{(5)}$, is discussed in depth in [23] and serves as a good example to illustrate the general splicing process discussed in V-C.



(a) Delay performance of the policies $\tilde{\pi}^{(3)}_{IQ}$, $\pi^{(3)}_{IQ}$, MaxWeight, $\pi^{(3)}_{TD}$ and $\pi^{(3)}_{BU}$ along the trajectory $\boldsymbol{\lambda}(s) = s \times [0.25, 0.74, 0.25]$, $s \in [0, 1]$, in the capacity region Λ_3.



(b) Delay performance of the policies $\tilde{\pi}^{(3)}_{IQ}$, $\pi^{(3)}_{IQ}$, MaxWeight, $\pi^{(3)}_{TD}$ and $\pi^{(3)}_{BU}$ along the trajectory $\boldsymbol{\lambda}(s) = s \times [0.74, 0.25, 0.74]$, $s \in [0, 1]$, in the capacity region Λ_3.

Fig. 9: Simulation results for the path-graph network with $N = 3$ for *Markovian* packet arrival processes. For all plots, and every $i \in \{1, 2, 3\}$ the transition probabilities of the arrival process (see Fig. 8) are chosen as follows $p_i = 0.10$, and $q_i = (\frac{1}{\lambda_i} - 1)p_i$.

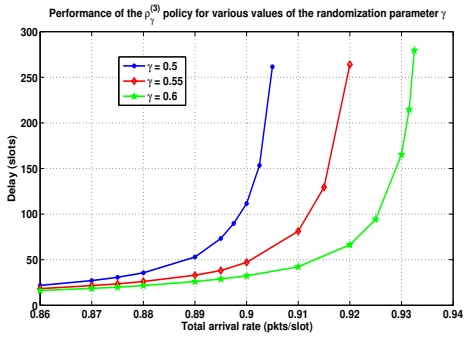


Fig. 10: Illustrating the loss of stability due to the "Flow-in-the-Middle" problem discussed in Sec. IV-D. We compare the delay performance of the policy $\rho^{(3)}_{\gamma}$ along the trajectory $\boldsymbol{\lambda}(s) = s \times [0.74, 0.25, 0.74]$, $s \in [0, 1]$, in the capacity region Λ_3. For every $s \leq 1$, the arrival rate vector lies within the interior of Λ_3 and is, hence, stabilizable. The policies can be seen to render the system unstable much before the system load parameter s hits 1.

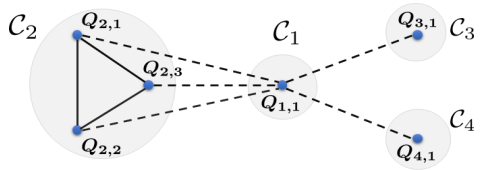


Fig. 11: The Star-of-Cliques (SoC) network used to study the performance of $\tilde{\phi}$. Simulation results are reported in Table III.

maximum size matching-QNB (MSM-QNB) policies on path-graphs with 3 queues and showed that this class contains stable, delay-optimal and even unstable policies. We then saw how priority policies for smaller path-graphs can be combined to construct QNB policies for larger networks. Next, we showed that policies so constructed are not MSM, but can be made MSM using a projection operator. We also showed how the delay properties of these MSM policies can be further improved by using certain observations of the nature of scheduling policies in $\tilde{\Pi}^{(N)}$. We then showed that there cannot exist QNB policies that are uniformly delay optimal over the entire capacity region, for any path graph network with $N \geq 4$ links.

Motivated by wireless networks commonly used for IoT-type applications, we introduced a new class of interference networks, called the "Cluster-of-Cliques" networks and studied two subclasses, namely, the *Star-of-Cliques* and the *Linear-Arrays-of-Cliques* networks. We then constructed QNB scheduling policies for both these classes, studied their stability and delay properties, and also developed a T.O. protocol that requires no explicit exchange of even occupancy information Our simulation results showed that the QNB policies we have developed perform better than existing scheduling policies that require complete knowledge of the system backlog in every slot.

In short, MaxWeight and policies based on it (such as MaxWeight-α) have been known to suffer from two major implementation issues, namely (i) disseminating queue length information across the network (or reporting it to some centralized scheduling entity), and (ii) finding the maximum weight independent set (MWIS), which for general conflict graphs, is famously an NP-hard problem. However, in the context of the current article, the latter problem is simplified. In fact, there exist dynamic programming approaches to solve the MWIS problem in linear time for path graphs. The outstanding issue in computing schedules, therefore, is one of information dissemination. Our work provides rigorous theoretical evidence that suggests that once the MWIS problem is simplified, detailed queue length information is (almost) irrelevant.

Future work will include extending these throughput-optimality results to non-Bernoulli arrival processes, and proving the throughput-optimality of $\pi^{(N)}_{TD}$ and $\pi^{(N)}_{BU}$ for general N-queue path graph networks[15]. Finally, we would also like

Cluster of Cliques Network	Mean sum queue length (packets)	
	QNB-MSM	MaxWeight
Star of Cliques (Fig. 11)	**45.535**	57.861
Linear Array of Cliques (Fig. 2a)	**245.038**	309.450

TABLE III: Comparison of sum queue length under the proposed Cluster of Cliques policies, and MaxWeight acting on the networks in Fig. 2a and Fig. 11. Details about the arrival rate vectors can be found in Sec. XI-V of the Appendix.

[15]The main bottleneck in the stability proofs appears to be combinatorial in

to explore such reduced state information based scheduling policies for more general conflict graphs and the existence of graphs that do not permit stable QNB scheduling policies.

REFERENCES

[1] L. Tassiulas and A. Ephremides, "Dynamic scheduling for minimum delay in tandem and parallel constrained queueing models," *Annals of Operations Research*, vol. 48, no. 4, pp. 333–355, 1994.

[2] M. Honrubia, "Industrial iot is booming thanks to a drop in sensor prices," 2017, https://www.ennomotive.com/industrial-iot-sensor-prices/.

[3] E. Dahlman, S. Parkvall, and J. Skold, *4G: LTE/LTE-advanced for mobile broadband*. Academic press, 2013.

[4] M. Alasti, B. Neekzad, J. Hui, and R. Vannithamby, "Quality of service in wimax and lte networks [topics in wireless communications]," *IEEE Communications Magazine*, vol. 48, no. 5, 2010.

[5] M. Lohstroh, H. Kim, J. C. Eidson, C. Jerad, B. Osyk, and E. A. Lee, "On enabling technologies for the internet of important things," *IEEE Access*, vol. 7, pp. 27 244–27 256, 2019.

[6] Z.-n. Kong, D. H. Tsang, B. Bensaou, and D. Gao, "Performance analysis of ieee 802.11 e contention-based channel access," *IEEE Journal on selected areas in communications*, vol. 22, no. 10, pp. 2095–2106, 2004.

[7] H. Wu, X. Wang, Q. Zhang, and X. Shen, "Ieee 802.11e enhanced distributed channel access (edca) throughput analysis," in *2006 IEEE International Conference on Communications*, vol. 1, 2006, pp. 223–228.

[8] H. Zhu and I. Chlamtac, "Performance analysis for ieee 802.11 e edcf service differentiation," *IEEE Transactions on wireless Communications*, vol. 4, no. 4, pp. 1779–1788, 2005.

[9] D. Dujovne, T. Watteyne, X. Vilajosana, and P. Thubert, "6tisch: deterministic ip-enabled industrial internet (of things)," *IEEE Communications Magazine*, vol. 52, no. 12, pp. 36–41, 2014.

[10] D. Dujovne, L. Grieco, M. Palattella, and N. Accettura, "6tisch on-the-fly scheduling draft-dujovne-6tisch-on-the-fly-04," *IETF 6TiSCH WG*, 2015.

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

[12] S. Sanghavi, D. Shah, and A. S. Willsky, "Message passing for maximum weight independent set," *IEEE Transactions on Information Theory*, vol. 55, no. 11, pp. 4822–4834, 2009.

[13] E. Modiano, D. Shah, and G. Zussman, "Maximizing throughput in wireless networks via gossiping," in *ACM SIGMETRICS Performance Evaluation Review*, vol. 34, no. 1. ACM, 2006, pp. 27–38.

[14] L. Jiang and J. Walrand, "Approaching throughput-optimality in distributed csma scheduling algorithms with collisions," *Networking, IEEE/ACM Transactions on*, vol. 19, no. 3, pp. 816–829, 2011.

[15] S. Rajagopalan, D. Shah, and J. Shin, "Network adiabatic theorem: an efficient randomized protocol for contention resolution," in *ACM SIGMETRICS Performance Evaluation Review*, vol. 37, no. 1. ACM, 2009, pp. 133–144.

[16] A. Mohan, A. Chattopadhyay, and A. Kumar, "Hybrid MAC protocols for low-delay scheduling," in *Mobile Ad Hoc and Sensor Systems (MASS), 2016 IEEE 13th International Conference on*. IEEE, 2016, pp. 47–55.

[17] D. Shah *et al.*, "Heavy traffic analysis of optimal scheduling algorithms for switched networks," *Annals of Applied Probability*, 2007.

[18] M. J. Neely, "Stochastic network optimization with application to communication and queueing systems," *Synthesis Lectures on Communication Networks*, vol. 3, no. 1, pp. 1–211, 2010.

[19] J. L. Gross and J. Yellen, *Graph theory and its applications*. CRC press, 2005.

[20] R. Diestel, "Graph theory. 2005," *Grad. Texts in Math*, vol. 101, 2005.

[21] B. Bollobás, *Modern graph theory*. Springer Science & Business Media, 2013, vol. 184.

[22] S. T. Maguluri, R. Srikant, and L. Ying, "Heavy traffic optimal resource allocation algorithms for cloud computing clusters," *Performance Evaluation*, vol. 81, pp. 20–39, 2014.

nature. Specifically, several of the 2^N realizations of $\boldsymbol{\zeta}(t)$ needed checking to establish stability. The proof for general N will, therefore, require a suitable counting argument that can reduce the number of realizations that need to be checked.

[23] A. Mohan, A. Gopalan, and A. Kumar, "Reduced-state, optimal scheduling for decentralized medium access control of wireless data collection networks," Tech. Rep., July 2019 https://drive.google.com/open?id=1zhXdE3TWRMtwrowBrVuem9CzfXrAZrmC.

[24] S. Hariharan and N. B. Shroff, "On sample-path optimal dynamic scheduling for sum-queue minimization in trees under the k-hop interference model," *IEEE/ACM Transactions on Networking*, vol. 24, no. 4, pp. 2458–2471, 2015.

[25] ——, "On sample-path optimal dynamic scheduling for sum-queue minimization in forests," *IEEE/ACM Transactions on Networking (TON)*, vol. 22, no. 1, pp. 151–164, 2014.

[26] N. McKeown, A. Mekkittikul, V. Anantharam, and J. Walrand, "Achieving 100% throughput in an input-queued switch," *IEEE Transactions on Communications*, vol. 47, no. 8, pp. 1260–1267, 1999.

[27] M. Garetto, T. Salonidis, E. W. Knightly, *et al.*, "Modeling per-flow throughput and capturing starvation in csma multi-hop wireless networks." in *Infocom*, 2006.

[28] C. Chaudet, I. G. Lassous, E. Thierry, and B. Gaujal, "Study of the impact of asymmetry and carrier sense mechanism in ieee 802.11 multi-hops networks through a basic case," in *Proceedings of the 1st ACM international workshop on Performance evaluation of wireless ad hoc, sensor, and ubiquitous networks*. ACM, 2004, pp. 1–7.

[29] B. Nardelli, J. Lee, K. Lee, Y. Yi, S. Chong, E. W. Knightly, and M. Chiang, "Experimental evaluation of optimal csma," in *INFOCOM, 2011 Proceedings IEEE*. IEEE, 2011, pp. 1188–1196.

[30] A. Warrier, S. Janakiraman, S. Ha, and I. Rhee, "Diffq: Practical differential backlog congestion control for wireless networks," in *INFOCOM 2009, IEEE*. IEEE, 2009, pp. 262–270.

[31] S. T. Maguluri, "Optimal scheduling algorithms for ad hoc wireless networks," 2011.

[32] X. Wang and K. Kar, "Throughput modelling and fairness issues in csma/ca based ad-hoc networks," in *INFOCOM 2005. 24th Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings IEEE*, vol. 1. Ieee, 2005, pp. 23–34.

[33] I. Rhee, A. Warrier, M. Aia, J. Min, and M. L. Sichitiu, "Z-MAC: a hybrid MAC for wireless sensor networks," *IEEE/ACM Transactions on Networking (TON)*, vol. 16, no. 3, pp. 511–524, 2008.

[34] P. Kinney, "The 802.15.4 CCA method," *Project: IEEE P802.15 Working Group for Wireless Personal Area Networks (WPANs)*, Submitted: Nov 14, 2001.

[35] A. L. Stolyar *et al.*, "Maxweight scheduling in a generalized switch: State space collapse and workload minimization in heavy traffic," *The Annals of Applied Probability*, vol. 14, no. 1, pp. 1–53, 2004.

[36] I. Keslassy and N. McKeown, "Analysis of scheduling algorithms that provide 100% throughput in input-queued switches," in *Proceedings of the Annual Allerton Conference on Communication Control and Computing*, vol. 39, no. 1. The University; 1998, 2001, pp. 593–602.

[37] V. M. G. Fayolle and M. Menshikov, *Topics in the Constructive Theory of Countable Markov Chains*. Cambridge University Press, 1995.

Avinash Mohan (S.M.'16-M'17) obtained his Ph.D. degree from the Indian Institute of Science, Bangalore, and the M.Tech. degree from the Indian Institute of Technology (IIT) Madras, both in Electrical Communication Engineering. He is currently a Postdoctoral Fellow at the Technion-Israel Institute of Technology, Haifa. His research interests include resource allocation in wireless communication networks, stochastic control and reinforcement learning.

Aditya Gopalan is an Assistant Professor and INSPIRE Faculty Fellow at the Indian Institute of Science, Electrical Communication Engineering. He received the Ph.D. degree in electrical engineering from The University of Texas at Austin, and the B.Tech. and M.Tech. degrees in electrical engineering from the Indian Institute of Technology Madras. He was an Andrew and Erna Viterbi Post-Doctoral Fellow at the Technion-Israel Institute of Technology. His research interests include machine learning and statistical inference, control, and algorithms for resource allocation in communication networks.

Anurag Kumar (B.Tech., Indian Institute of Technology (IIT) Kanpur; PhD, Cornell University, both in Electrical Engineering) was with Bell Labs, Holmdel, N.J., for over 6 years. Since then he has been on the faculty of the ECE Department at the Indian Institute of Science (IISc), Bangalore; he is at present the Director of the Institute. His area of research is communication networking, and he has recently focused primarily on wireless networking. He is a Fellow of the IEEE, the Indian National Science Academy (INSA), the Indian National Academy of Engineering (INAE), and the Indian Academy of Sciences (IASc). He was an associate editor of IEEE Transactions on Networking, and of IEEE Communications Surveys and Tutorials.

XI. Appendix

A. Glossary of Acronyms

- BU: Bottom-up
- CoC: Cluster-of-Cliques
- CSMA: Carrier Sense Multiple Access
- D.O.: Uniformly Delay Optimal (see Defn. IV-A).
- IID: Independent and Identically Distributed
- IoT: Internet of Things
- LAOC: Linear-Array-of-Cliques
- MAC: Medium Access Control
- MSM: Maximum Size Matching
- MW : The MaxWeight algorithm defined in [11].
- $MW\alpha$: The MaxWeight-α algorithm defined in [35]
- MWIS: Maximum Weight Independent Set
- QNB: Queue Nonemptiness Based (policy)
- SoC: Star-of-Cliques
- T.O.: Throughput Optimal
- TD: Top-down.

B. Glossary of Notation

1) $\mathbb{I}_{\{\text{CONDITION}\}}$: the indicator function, which evaluates to 1 whenever CONDITION is true, and 0 otherwise.
2) $\boldsymbol{\zeta}(t)$: the occupancy vector or nonemptiness vector at time t, defined as $\boldsymbol{\zeta}(t) = \left[\mathbb{I}_{\{Q_1(t)>0\}}, \cdots, \mathbb{I}_{\{Q_N(t)>0\}}\right]$.
3) $\mathcal{V}$: is the set of all activation vectors. Clearly, in a system of N queues, $\mathcal{V} \subsetneq \{0,1\}^N$ due to interference constraints.
4) Λ_N : The capacity region of a path graph interference network with N queues (communication links).
5) $\sigma(X)$: The sigma algebra generated by the random variable X.
6) $\Pi^{(N)}$: the class of all scheduling policies defined on path graphs.
7) $\Gamma_M^{(N)}$: the class of all Maximum Size Matching (MSM) policies.
8) $\Pi_M^{(N)}$: the class of all policies that take only the occupancy vector $\boldsymbol{\zeta}(t)$ as input and activate the largest number of non empty queues in every slot, .i.e., MSM policies that require only the empty or nonempty status of the queues in the network.
9) $\tilde{\Pi}^{(N)}$: the class of all MSM policies within $\Pi_M^{(N)}$ that additionally break ties in favour of inner queues (see condition 2).
10) $\mid A \mid$: represents the cardinality of set A.
11) $Geo(\lambda)/Geo(\mu)/1$ queue: A queue with a Bernoulli arrival process whose interarrival periods are geometrically distributed with mean λ, and whose service times are IID and geometrically distributed with mean μ.
12) $\pi_{TD}^{(N)}$ and $\pi_{BU}^{(N)}$: Top-Down and Bottom-Up policies for path graph networks with N queues.
13) $\pi_{SP}^{(2N-1)}$: The policy obtained by *splicing* $\pi_{TD}^{(N)}$ and $\pi_{BU}^{(N)}$. This policy is not MSM.
14) $\{\tilde{\pi}_i^{(4)}, 1 \le i \le 4\}$: These are the four policies within the class $\tilde{\Pi}^{(4)}$ that are both MSM and break ties in favor of the inner queues, i.e., Queues 2 and 3.
15) $\pi_{TI}^{(4)}$: The policy on 4-queue path graphs obtained by splicing $\pi_{TD}^{(3)}$ and $\pi_{IQ}^{(3)}$, the Top-Down and Bottom-Up policies on 3-queue path graphs. Since it is not obtained by splicing Top-Down and Bottom-Up policies, we do not give it the subscript "SP."
16) $\mathcal{N}$: The total number of queues in an Linear Array of Cliques (LAoC) or Star-of-Cliques (SoC) network.
17) $\phi_{IC}^{(S)}$: The policy defined on Star-of-Cliques (SoC) networks that prioritizes the inner clique over all the peripheral cliques. It is defined in Sec. VII-A.
18) $\tilde{\phi}_{IC}^{(S)}$ The "S" in the superscript stands for SoC network, and "IC" in the subscript shows that they prioritize the *inner clique,* i.e., C_1. It is defined in Sec. VII-A.
19) $\theta_{TD}^{(3L)}$ and $\theta_{BU}^{(3L)}$: Top-Down and Bottom-Up policies defined over LAoC networks with 3 cliques. Here, the "L" in the superscript stands for LAoC network.

    Remark. Throughout Sec. VII, ϕ will always represent a policy for SoC networks and θ for LAoC network.
20) $\phi_{IC}^{(S)}(T)$: The version of $\phi_{IC}^{(S)}$ defined, once again for SoC networks, that requires knowledge of the vector $\boldsymbol{\zeta}$ only every T time slots.
21) $\phi_{CS}^{(S)}$: that, like the QZMAC protocol in [16], takes scheduling decisions based solely on the information gathered by sensing the channel for activity. Obviously, the "CS" in the subscript stands for channel sensing.

C. Throughput Optimality of Queue Nonemptiness-based Scheduling in Fully Connected Graphs

The proof of Thm. 1, i.e., throughput-optimality of policies satisfying property $\mathcal{P}$, proceeds via a Lyapunov argument. The sole purpose of this this subsection is to provide some intuition to the reader about how we came to construct the Lyapunov function used therein. This subsection, therefore, is not necessary to understand the proof and may be skipped without loss of continuity.

Consider stabilizing a collocated network of N queues described by a *Fully Connected* interference graph. Here the capacity region consists of all rate vectors $\boldsymbol{\lambda} \in \mathbb{R}_+^N$ that satisfy $\sum_{i=1}^N \lambda_i < 1$. Any policy that schedules a nonempty queue in every slot (if there exists one) is T.O. To see this, define $Q(t) := \sum_{i=1}^N Q_i(t)$, $A(t+1) = \sum_{i=1}^N A_i(t+1)$ and $D(t) := \sum_{i=1}^N D_i(t)$ and consider the Lyapunov Function

$$L(\mathbf{Q}(t)) := \left(\sum_{i=1}^N Q_i(t)\right)^2 = Q^2(t). \tag{15}$$

Using the fact that for any three non negative reals x, y, z, $((x-y)^+ + z)^2 \le x^2 + y^2 + z^2 - 2x(y-z)$, we see that

$$Q^2(t+1) \le Q^2(t) + D^2(t) + A^2(t) - 2Q(t)\left(D(t) - A(t)\right).$$

Hence, the expected single slot drift becomes

$$\begin{aligned}
&\mathbb{E}\left[L(\mathbf{Q}(t+1)) - L(\mathbf{Q}(t)) \mid \mathbf{Q}(t) = \mathbf{q}\right] \\
&\overset{\dagger 1}{\le} \mathbb{E}\left[D^2(t) + A^2(t) \mid \mathbf{q}\right] - 2q\mathbb{E}\left[D(t) - A(t) \mid \mathbf{q}\right] \\
&\overset{\dagger 2}{=} 1 + N^2 - 2q\left(\mathbb{E}\left[D(t) \mid \mathbf{q}\right] - \sum_{i=1}^N \lambda_i\right),
\end{aligned} \tag{16}$$

where in †1, $\mathbf{q} = [q_1, \cdots, q_N]$, and $q = \sum_{i=1}^{N} q_i$. In †2 we have used the fact that $D(t) \leq 1$ since at most one queue can be served per slot, $A(t) \leq N$ since at most one packet can arrive in each of the N queues in a slot, and because arrivals are independent of the current system state, $\mathbb{E}\Big[A(t) \mid q\Big] = \mathbb{E}A(t) = \sum_{i=1}^{N} \lambda_i$. Since the policy schedules a non empty queue in every slot,

$$\begin{aligned} \mathbb{E}\Big[D(t) \mid q\Big] &= \mathbb{I}_{\{Q(t)>0\}}, \text{ i.e.,} \\ \mathbb{E}\Big[\sum_{i=1}^{N} D_i(t) \mid q\Big] &= \mathbb{I}_{\{\sum_{i=1}^{N} Q_i(t)>0\}} \end{aligned} \tag{17}$$

Taking expectations on both sides of (16), we see that

$$\begin{aligned} \mathbb{E}Q^2(t+1) - \mathbb{E}Q^2(t) \leq 1 + N^2 - 2\mathbb{E}\left(Q(t)\mathbb{I}_{\{Q(t)>0\}}\right) \\ -\mathbb{E}Q(t)\left(\sum_{i=1}^{N} \lambda_i\right), \end{aligned}$$

and since $Q(t) \geq 0, \ \forall t, \ \mathbb{E}\left(Q(t)\mathbb{I}_{\{Q(t)>0\}}\right) = \mathbb{E}Q(t)$. Setting $\epsilon = 1 - \sum_{i=1}^{N} \lambda_i$ ($\epsilon > 0$ by stability considerations) we get

$$\mathbb{E}Q^2(t+1) - \mathbb{E}Q^2(t) \leq 1 + N^2 - 2\epsilon\mathbb{E}Q(t). \tag{18}$$

Summing the above over $t = 0, 1, \cdots, T-1$, we get

$$\mathbb{E}Q^2(T) - \mathbb{E}Q^2(0) \leq T(1+N^2) - 2\epsilon \sum_{t=0}^{T-1} \mathbb{E}Q(t). \tag{19}$$

A little bit of algebra shows that

$$\begin{aligned} \frac{1}{T}\sum_{t=0}^{T-1} \mathbb{E}Q(t) &\leq \frac{1+N^2}{2\epsilon} + \frac{\mathbb{E}Q^2(0)}{2\epsilon T} \\ \Rightarrow \limsup_{T\to\infty} \frac{1}{T}\sum_{t=0}^{T-1} \mathbb{E}Q(t) &< \infty, \end{aligned}$$

which implies strong stability. In the sequel, we will call this rather standard technique [18] of showing strong stability, the *telescoping sum method.* As a precursor to the proof of Lem. 1, observe that this class of policies satisfies property $\mathcal{P}$ in the lemma, i.e., ($\mathcal{P}$) since $\mathbb{E}\Big[D(t) \mid q\Big] = \mathbb{I}_{\{Q(t)>0\}}$ means that $\sum_{i=1}^{N} D_i(t) = 0 \iff \sum_{i=1}^{N} Q_i(t) = 0$. ■

D. Proof of Lem. 1

We define a Lyapunov function $L(t) : \mathbb{N}^N \to \mathbb{R}_+$ as

$$L(\mathbf{Q}(t)) := \sum_{i=1}^{N-1} (Q_i(t) + Q_{i+1}(t))^2 \tag{20}$$

With a slight abuse of notation, we denote $L(\mathbf{Q}(t))$ simply by $L(t)$. Using the Lyapunov drift argument and the telescoping sum method used in Sec. XI-C, we now show how property $\mathcal{P}$ ensures *strong stability* of the system when the arrivals lie in Λ^o. To simplify notation, we denote $D_i(t)$ and $A_i(t+1)$ by D_i and A_i respectively, for every i and let $\mathbf{Q} = [Q_1, \ldots, Q_N]$.

$$\begin{aligned} &\mathbb{E}\left[L(t+1) - L(t) \mid \mathbf{Q}(t) = \mathbf{Q}\right] \\ &= \sum_{i=1}^{N-1} \mathbb{E}\Big[(Q_i - D_i + A_i + Q_{i+1} - D_{i+1} + A_{i+1})^2 \\ &\quad -(Q_i + Q_{i+1})^2 \mid \mathbf{Q}(t) = \mathbf{q}\Big] \\ &\overset{*}{\leq} \sum_{i=1}^{N-1} \Big[(Q_i + Q_{i+1})^2 + 1 + 4 - 2(Q_i + Q_{i+1}) \\ &\quad (\mathbb{E}\left[D_i + D_{i+1} \mid \mathbf{Q}\right] - \lambda_i - \lambda_{i+1}) - (Q_i + Q_{i+1})^2\Big] \\ &= \sum_{i=1}^{N-1} \Big[5 - 2(Q_i + Q_{i+1}) \\ &\quad (\mathbb{E}\left[D_i + D_{i+1} \mid \mathbf{Q}(t) = \mathbf{Q}\right] - \lambda_i - \lambda_{i+1})\Big], \end{aligned} \tag{21}$$

where in inequality $*$, firstly, we have used the fact that for any 3 reals x, y, z, $(x - y + z)^2 \leq x^2 + y^2 + z^2 - 2x(y - z)$ and set $x = q_i + q_{i+1}$, $y = D_i + D_{i+1}$ and $z = A_i + A_{i+1}$. We then use the fact that $D_i + D_{i+1} \leq 1$ in any time slot due to the scheduling constraints and since all arrivals are Bernoulli, $A_i + A_{i+1} \leq 2$, for all $1 \leq i \leq N-1$. Taking expectation on both sides of Eqn. (21), and thus *removing conditioning,* we get

$$\begin{aligned} \mathbb{E}\left[L(t+1) - L(t)\right] \leq \sum_{i=1}^{N-1} \Big[5 - 2\mathbb{E}\left((Q_i + Q_{i+1})\right. \\ \left.\times \mathbb{E}\left[D_i + D_{i+1} \mid \mathbf{Q}(t)\right]\right) \\ +2(\lambda_i + \lambda_{i+1})\mathbb{E}\left(Q_i + Q_{i+1}\right)\Big] \end{aligned} \tag{22}$$

We now use the fact that the policy satisfies property $\mathcal{P}$, to see that $\mathbb{E}\left[D_1 + D_2 \mid \mathbf{Q}(t)\right] = \mathbb{I}_{\{Q_i + Q_{i+1} > 0\}}$, $w.p.1$, and the fact that for any non negative random variable Z, $\mathbb{E}\left(Z\mathbb{I}_{\{Z>0\}}\right) = \mathbb{E}Z$, whereby,

$$\mathbb{E}\left((Q_i + Q_{i+1})\mathbb{E}\left[D_i + D_{i+1} \mid \mathbf{Q}(t)\right]\right) = \mathbb{E}(Q_i + Q_{i+1}).$$

This means that

$$\begin{aligned} \mathbb{E}\left((Q_i + Q_{i+1})\left(\mathbb{E}\left[D_i + D_{i+1} \mid \mathbf{Q}(t)\right] + (\lambda_i + \lambda_{i+1})\right)\right) \\ = \epsilon_i \mathbb{E}(Q_i + Q_{i+1}), \end{aligned}$$

where $\epsilon_i = 1 - \lambda_i - \lambda_{i+1}$. Note that from the definition of Λ^o, $\epsilon_i > 0, \ \forall 1 \leq i \leq N-1$. Substituting this in Eqn. (22), we get

$$\begin{aligned} &\mathbb{E}\left[L(t+1) - L(t)\right] \\ &\leq 5(N-1) - 2\sum_{i=1}^{N-1} \epsilon_i \mathbb{E}\left[Q_i(t) + Q_{i+1}(t)\right], \\ &\overset{\dagger}{\leq} 5(N-1) - 2\sum_{i=1}^{N-1} \epsilon \mathbb{E}\left[Q_i(t) + Q_{i+1}(t)\right], \\ &= 5(N-1) - 2\epsilon\mathbb{E}Q_1(t) - 4\epsilon\sum_{i=2}^{N-2} \mathbb{E}Q_i(t) - 2\epsilon\mathbb{E}Q_N(t), \end{aligned}$$

where in inequality †, $\epsilon := \min_{1\le i\le N-1}\epsilon_i$. Since $\epsilon > 0$, $-4\epsilon\sum_{i=2}^{N-2}\mathbb{E}Q_i(t) < -2\epsilon\sum_{i=2}^{N-2}\mathbb{E}Q_i(t)$. Using this, we get

$$\mathbb{E}\left[L(t+1)-L(t)\right] \quad\le\quad 5(N-1)-2\epsilon\sum_{i=1}^{N}\mathbb{E}Q_i(t). \quad (23)$$

Using the telescoping sum technique (see [18]) it can now be shown that the process $\{\mathbf{Q}(t),\ t\ge 0\}$, is strongly stable.■

E. Proof of Thm. 2

Queues 1 and 2 form a priority queue and $\pi^{(3)}_{TD}$ serves the pair of queues 1 and 2 whenever either of them is nonempty. So, $\pi^{(3)}_{TD}$ satisfies Property $\mathcal{P}$ for $i = 1$ (specifically, Eqn. ($\mathcal{P}$)), which means that the process $\{[Q_1(t), Q_2(t)],\ t\ge 0\}$, is strongly stable. Further, since Queue 1 receives the highest priority, as soon as a packet arrives it is served and leaves the system at the end of the slot. Consequently, Queue 1 behaves like a $Geo(\lambda_1)/D/1$ queue, with service time being exactly one slot. This also means that by starting out with $Q_1(0)\le 1$, in any time slot, Queue 1 has at most 1 packet, which is the arrival during that slot, i.e., $Q_i(t) = A_i(t),\ \forall t\ge 1$. Also, $P\{Q_i(t) > 0\} = \lambda_1$.

Queues 1 and 2 form a priority queueing system. This means that the packet at the Head of Line (HOL) position in Queue 2 is served whenever $Q_1(t) = 0$. Since $Q_1(t) = A_1(t)$, $P\{Q_1(t) = 0\} = 1-\lambda_1$ independently of $Q_2(t)$. Moreover, the arrivals to Queue 1 are Bernoulli with mean λ_1, which means that the service time B_2 of every packet in Queue 2 is IID and geometrically distributed with mean $\frac{1}{1-\lambda_1}$. Specifically, $B_2 \sim Geo(\frac{1}{1-\lambda_1})$. This means that Queue 2 behaves like a (refer glossary XI-B for an explanation of this notation) $Geo(\lambda_1)/Geo(\frac{1}{1-\lambda_1})/1$ queue, and since $\lambda_2 < (1-\lambda_1)$, Queue 2 is stable. Furthermore, $\{[Q_1(t), Q_2(t)], t\ge 0\}$ forms an aperiodic, irreducible positive recurrent DTMC. This means that there exists a steady state probability measure on Queue 2's backlog such that,

$$\begin{aligned}\lim_{t\to\infty}P\{Q_2(t)=0\} &= 1-\lambda_2\mathbb{E}B_2\\ &= 1-\frac{\lambda_2}{1-\lambda_1}. \end{aligned}\quad (24)$$

From the definition of $\pi^{(3)}_{TD}$ we see that Queue 3 is scheduled for service whenever either Queue 1 is nonempty or when *both* Queue 1 and Queue 2 are empty. Specifically, the choice of the activation set is completely governed by the backlogs of queues 1 and 2 and does not depend on Queue 3 at all. This means the service given to Queue 3 in every slot is independent of its backlog in that slot. Suppose we begin both Queue 1 and Queue 2 in their steady state distributions,

$$\begin{aligned}P\{S_3(t)=1\} &= P\{Q_1(t)>0\}+P\{Q_1(t)=0,Q_2(t)=0\}\\ &= \lambda_1+P\{Q_2(t)=0\}P\{Q_1(t)=0\mid Q_2(t)=0\}\\ &\overset{*1}{=} \lambda_1+\left(1-\frac{\lambda_2}{1-\lambda_1}\right)P\{A_1(t)=0\mid Q_2(t)=0\}\\ &\overset{*2}{=} \lambda_1+\left(1-\frac{\lambda_2}{1-\lambda_1}\right)(1-\lambda_1)\\ &= 1-\lambda_2>\lambda_3. \end{aligned}\quad (25)$$

Equality $*1$ uses Eqn. (24) and $*2$ uses the fact that external arrivals to Queue 2 in a slot are independent of the backlog of Queue 2 in that slot. To show that Queue 3 is strongly stable, define $L:\mathbb{N}\to\mathbb{R}_+$ as $L(Q_3(t)) = Q_3^2(t)$.

$$\begin{aligned}\mathbb{E}\left[L(t+1)-L(t)\right] &= \mathbb{E}[Q_3^2(t+1)-Q_3^2(t)]\\ &= \mathbb{E}[\left((Q_3(t)-S_3(t))^+ + A_3(t+1)\right)^2 - Q_3^2(t)]\\ &\overset{\star 3}{\le} 2-2\mathbb{E}Q_3(t)\mathbb{E}[S_3(t)-A_3(t+1)]\\ &= 2-2\mathbb{E}Q_3(t)\left(1-\lambda_2-\lambda_3\right)\\ &\overset{\star 4}{=} 2-2\delta\mathbb{E}Q_3(t) \end{aligned}\quad (26)$$

In $\star 3$, we have once again used the fact that for any four non negative reals w, x, y and z, with $w\le(x-y)^+ + z$, $w^2\le x^2+y^2+z^2-2x(y-z)$, with $w = Q_3(t+1), x = Q_3(t), y = S_3(t)$ and $z = A_3(t+1)$. Further, $S_3(t)\le 1$ and $A_3(t+1)\le 1$. Finally, in $\star 4$, $\delta = 1-\lambda_2-\lambda_3 > 0$ by capacity constraints. This shows that Queue 3 is also strongly stable, and since

$$\begin{aligned}&\limsup_{T\to\infty}\frac{1}{T}\sum_{t=0}^{t-1}\sum_{i=1}^{3}\mathbb{E}_{\pi^{(3)}_{TD}}Q_i(t)\\ &\le \limsup_{T\to\infty}\frac{1}{T}\sum_{t=0}^{t-1}\mathbb{E}_{\pi^{(3)}_{TD}}Q_1(t)+\limsup_{T\to\infty}\frac{1}{T}\sum_{t=0}^{t-1}\mathbb{E}_{\pi^{(3)}_{TD}}Q_2(t)\\ &+\limsup_{T\to\infty}\frac{1}{T}\sum_{t=0}^{t-1}\mathbb{E}_{\pi^{(3)}_{TD}}Q_3(t),\end{aligned}$$

the system is also strongly stable under this policy. The other policy $\pi^{(3)}_{BU}$ simply swaps the priorities of Queues 1 and 3, and its proof proceeds as before, mutatis mutandis. ■

F. Proof of Thm. 3

Since, by definition, for any Queue i, $D_i(t) = S_i(t)\mathbb{I}_{\{Q_i(t)>0\}}$, $Q_1(t)+Q_2(t) = 0$ always means $D_1(t)+D_2(t) = 0$. To show the converse, we consider several cases

- $Q_1(t) > 0$ and $Q_2(t) > 0$ means that either in step 2 or 3 of MSM, on of these queues will get scheduled and either $D_1(t) = 1$ or $D_2(t) = 1$.
- $Q_1(t) = 0$ and $Q_2(t) > 0$ means that MSM schedules Queue 2 in step 2, and $D_2(t) = 1$.
- $Q_2(t) = 0$ and $Q_1(t) > 0$ means that MSM schedules Queue 1 in either step 1 or step 3, depending on the length of Queue 3, whereby $D_1(t) = 1$.

Following the same logic, we state a similar result for $D_2(t)+D_3(t)$. This means that $\tilde{\pi}^{(3)}_{IQ}$ satisfies property $\mathcal{P}$, and from Lem. 1, $\tilde{\pi}^{(3)}_{IQ}$ is T.O. ■

G. Proof of Thm. 4

We adapt the technique used in Lem. 4.2 of [1] to prove this result[16]. The main idea is to construct a sequence $\{\pi'_k, k\ge 0\}$ of intermediate policies such that the backlog in every queue converges sample path-wise to that of $\tilde{\pi}^{(3)}_{IQ}$ which means that for every $t\ge 0$,

$$\lim_{k\to\infty}Q_l^{\pi'_k}(t) = Q_l^{\tilde{\pi}^{(3)}_{IQ}}(t),\ 1\le l\le 3, \quad (27)$$

[16]The reader should note that there is a *typo* in [1] that labels two results as Lem 4.1. Here, we refer to the latter as 4.2 to avoid confusion.

over every sample path. Each policy in the sequence π'_k is designed to provide smaller sum queue length than its predecessor π'_{k-1} and the chosen policy π. Towards this end, we first couple arrivals to the systems on which $\tilde{\pi}^{(3)}_{IQ}$, $\{\pi'_k,\ k \geq 0\}$ and π act (by assumption, the initial conditions are equal, i.e., on every sample path $\mathbf{Q}^{\pi}(0) = \mathbf{Q}^{\tilde{\pi}^{(3)}_{IQ}}(0) = \mathbf{Q}^{\pi'_k}(0) = \mathbf{Q},\ \forall k \geq 0$, where $\mathbf{Q} \in \mathbb{N}^3$ is some generic queue length vector). We then define π'_0 as follows. In slot 0, π'_0 follows $\tilde{\pi}^{(3)}_{IQ}$ which means the activation vectors chosen by the two policies are the same. In other words, $\mathbf{s}^{\pi'_0}(0) = \mathbf{s}^{\tilde{\pi}^{(3)}_{IQ}}(0)$, with the superscripts denoting the policy. We now show that at $t = 1$, the following conditions are satisfied by $\mathbf{Q}^{\pi'_0}(1)$ and $\mathbf{Q}^{\pi}(1)$. In Condition 4 below, the indices j_k are as defined in Lem. 19.

1) $Q_l^{\pi'_0}(t) \leq Q_l^{\pi}(t) + 1$ for $1 \leq l \leq N$. Here, $N = 3$.
2) If $Q_l^{\pi'_0}(t) = Q_l^{\pi}(t)+1$ and $l < N$ then $Q_{l+1}^{\pi'_0}(t) = Q_{l+1}^{\pi}(t)-1$.
3) If $Q_l^{\pi'_0}(t) = Q_l^{\pi}(t)+1$ and $l > 1$ then $Q_{l-1}^{\pi'_0}(t) = Q_{l-1}^{\pi}(t)-1$.
4) If $j_1 = 1$, $j_2 = N$ and N is odd, then $Q_1^{\pi'_0}(t) \leq Q_1^{\pi}(t)$ and $Q_3^{\pi'_0}(t) \leq Q_3^{\pi}(t)$.

The first condition is obvious since at most one packet can depart from a queue in a slot and since arrivals to the two systems are coupled. For the second condition observe that since arrivals are coupled, $Q_l^{\pi'_0}(1) = Q_l^{\pi}(1) + 1 \Rightarrow Q_l^{\pi'_0}(0) = Q_l^{\pi}(0) = Q_l(0) \neq 0$, and $s_l^{\pi}(0) = 1$ while $s_l^{\pi'_0}(0) = 0$, i.e., Queue l was not empty at 0, and π served it while π'_0 did not. This is because if either the queue was empty at time 0 or both the policies served it, the 1 packet mismatch would never have occurred. We now have two cases.

- $l = 1 \Rightarrow Q_2(0) \neq 0$, and, $Q_3(0) = 0$, since by the definition, $\tilde{\pi}^{(3)}_{IQ}$ ignores the extreme queues when they are nonempty only when Queue 2 is nonempty and $Q_1(t) \cdot Q_3(t) = 0$. Hence, π'_0 serves Queue 2 in slot 0 while π does not, resulting in $Q_2^{\pi'_1}(0) = Q_2^{\pi}(0) - 1$.
- $l = 2 \Rightarrow Q_1(0) \neq 0$ and $Q_3(0) \neq 0$, since by definition, $\tilde{\pi}^{(3)}_{IQ}$ ignores Queue 2 queues when it is nonempty only when both Queue 1 and Queue 3 are nonempty. In this case π'_0 serves Queue 1 and Queue 3 in slot 0 while π does not, resulting in $Q_1^{\pi'_1}(1) = Q_1^{\pi}(1) - 1$ and $Q_3^{\pi'_0}(1) = Q_3^{\pi}(1) - 1$.

The third condition is explained in a similar manner and follows easily from symmetry. When $j_1 = 1$ and $j_2 = 3$, all three queues are nonempty and π' serves both. This proves the fourth condition. When these 4 conditions hold, the sum backlog with π' is not larger than with π due to the following reason. When all queues are initially nonempty (meaning $Q_l(0) > 0, 1 \leq l \leq 3$), this is true from condition 4. When only two adjacent queues are nonempty, conditions 2 and 3, as the case may be, ensure this. When only queues 1 and 3 are nonempty, $\tilde{\pi}^{(3)}_{IQ}$ and hence, π'_0 serve both of them. The case with only one nonempty queue at $t = 0$ is trivial. Thus,

$$\sum_{i=1}^{3} Q_i^{\pi'_0}(1) \leq \sum_{i=1}^{3} Q_i^{\pi}(1).$$

For $t \geq 1$, the definition of π'_0 and the rest of the proof of how the above inequality is ensured at every $t \geq 0$ is the same as in [1] and will not be repeated. For every $k \geq 0$, π'_{k+1} is defined as the policy that follows $\pi^{(3)}_{OQ}$ over slots $0, 1, \ldots, k$ and over $k+1, \ldots,$ is defined as in [1] so as to satisfy

$$\sum_{i=1}^{3} Q_i^{\pi_{k+1}}(t) \leq \sum_{i=1}^{3} Q_i^{\pi_k}(t),\ \forall t \geq 0,\ \forall k \geq 1. \tag{28}$$

Again, by construction, it is clear that

$$\lim_{k\to\infty} Q_i^{\pi'_k}(t) \stackrel{s}{=} Q_i^{\tilde{\pi}^{(3)}_{IQ}}(t),\ \forall t \geq 0,\ 1 \leq i \leq N, \tag{29}$$

where $\stackrel{s}{=}$ means over every sample path (and is stronger than "a.s."). Eqn. 28 together with Eqn. 29 give us 11. This completes the proof. ■

H. Proof of Prop. 5

Consider real numbers ϵ and δ, such that $\delta > 0, 0 < \epsilon < 0.5$ and $\epsilon + \delta \leq 0.5$. Let $\boldsymbol{\lambda} = [0.5 - \epsilon - \delta, 0.5 + \epsilon, 0.5 - \epsilon - \delta]$. Clearly, $\boldsymbol{\lambda} \in \Lambda^o$. At time t, define the event $S_2 := \{\text{Queue 2 is served in slot } t\}$ and let $\mathbf{Q}(t) = [q_1, q_2, q_3]$.

$$\begin{aligned}
P\{S_2 = 1\} &= P\{S_2 = 1 \mid q_1 + q_3 > 0\}P\{q_1 + q_3 > 0\} \\
&\quad + P\{S_2 = 1 \mid q_1 + q_3 = 0\}P\{q_1 + q_3 = 0\}. \\
&\leq 0 \cdot P\{q_1 + q_3 > 0\} + 1 \cdot P\{q_1 + q_3 = 0\} \\
&= 1 - P\{q_1 + q_3 > 0\} \\
&= 1 - P\{q_1 > 0 \text{ or } q_3 > 0\} \\
&\leq 1 - P\{A_1(t) > 0 \text{ or } A_3(t) > 0\} \\
&= P\{A_1(t) = A_3(t) = 0\} = (1 - (0.5 - \epsilon - \delta))^2 \\
&= (0.5 + \epsilon + \delta)^2
\end{aligned} \tag{30}$$

If $x := (0.5 + \epsilon) = \lambda_2$, to prove the instability of $\pi^{(3)}_{OQ}$ one only needs to solve the nonlinear program (31) below. That will establish that $P\{S_2(t) = 1\} < \lambda_2$, and hence prove that Queue 2 is unstable.

$$\begin{aligned}
\textit{Find} \quad & (x + \delta)^2 < x, \\
s.t. \quad & \delta > 0, \\
& x > 0.5, \\
& x < 1, \\
& x + \delta \leq 1.
\end{aligned} \tag{31}$$

The problem above is easily solved, for example, by $x = 0.75$, from which we conclude that $\pi^{(3)}_{OQ}$ is unstable. ■

I. Proof of Prop. 7

1) Brief Digression: A Sufficient Condition for MSM Policies : Before stating the proof of the theorem, we would like to restate Lem. 4.1 in [1] for the reader's convenience. We will be invoking this result in multiple proofs later in the paper. It is important to note that the lemma below is only a *sufficient* condition for a policy to be MSM.

Given an occupancy vector $\boldsymbol{\zeta}$, let $k = k(\boldsymbol{\zeta})$ be the *twice* the number of runs of nonempty queues, and $j_1 = j_1(\boldsymbol{\zeta}), \ldots, j_k = j_k(\boldsymbol{\zeta})$, the nonempty queues that mark the beginnings ($j_{\{odd\ subscript\}}$) and ends ($j_{\{even\ subscript\}}$) of

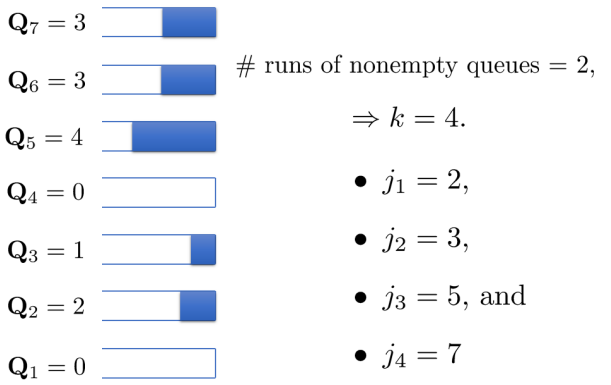


Fig. 12: Figure depicting how runs of non empty queues are numbered. Here, $N = 7$ and since $\boldsymbol{\zeta} = [0,1,1,0,1,1,1]^T$, $S(\boldsymbol{\zeta}) = \{[0,1,1,0,0,1,0,1]^T, [0,0,1,0,1,0,1,0,1]^T\}$. There are two runs of nonempty queues, the first beginning at Queue 2 and ending at Queue 3, and the second beginning at Queue 5 and ending at Queue 7. Hence, $j_1 = 2, j_2 = 3, j_3 = 5$ and $j_5 = 7$. Notice that odd subscripts indicate the *beginning* of these runs, while even subscripts indicate their ends.

the nonempty runs, or the two extreme queues (Queues 1 and N). Fig. 12 illustrates this numbering scheme. Clearly,

- If $j > j_k$ or $j < j_1$ Queue j is empty,
- If $j_{2m-1} \le j \le j_{2m}$, $m = 1, 2, \ldots, \frac{k}{2}$, Queue j is nonempty, and
- If $j_{2m} \le j \le j_{2m+1}$, $m = 1, 2, \ldots, \frac{k}{2} - 1$, Queue j is empty.

Lemma 19. (Lem. 4.1 in [1]) $\mathbf{S}(t) \in S(\boldsymbol{\zeta}(t))$ if

1) **Odd-length run condition:** If $j_{2m} - j_{2m-1}$ is even, then for all $j_{2m-1} \le j \le j_{2m}$,
$$S_j(t) = \begin{cases} 1 \text{ if } j - j_{2m-1} \text{ is even,} \\ 0 \text{ otherwise,} \end{cases}$$
$m = 1, \ldots, k/2$.
2) **Even-length run condition:** If $j_{2m} - j_{2m-1}$ is odd, then any one of the following 3 conditions must be satisfied
   a) For every $j_{2m-1} \le j \le j_{2m}$,
   $$S_j(t) = \begin{cases} 1 \text{ if } j - j_{2m-1} \text{ is even,} \\ 0 \text{ otherwise,} \end{cases}$$
   b) For every $j_{2m-1} \le j \le j_{2m}$,
   $$S_j(t) = \begin{cases} 1 \text{ if } j - j_{2m-1} \text{ is odd,} \\ 0 \text{ otherwise,} \end{cases}$$
   or,
   c) There exists an l such that
   $$S_j(t) = \begin{cases} 1 \text{ if } j - j_{2m-1} \text{ is even and } j_{2m-1} \le j < l, \\ \quad \text{or } j_{2m} - j \text{ is even and } j_{2m} \ge j > l + 1, \\ 0 \text{ otherwise,} \end{cases}$$
   $m = 1, \ldots, k/2$.
3) **Inner queues priority condition:** If $j_1 = 1$,
$$S_j(t) = \begin{cases} 1 \text{ if } j_2 - j \text{ is even, } j_1 \le j \le j_2, \\ 0 \text{ otherwise,} \end{cases}$$
and similarly for the case with $j_2 = N$.

It is easily seen that condition 3 above is *not necessary*, and any $\mathbf{S}$ that satisfies the first two will automatically exist in $S(\boldsymbol{\zeta})$.

Lem. 19 gives a sufficient condition for an activation vector to be of maximum size. We will now show that in every slot t, the activation vector $\mathbf{s}(t)$ that $\pi_{TD}^{(N)}$ produces satisfies this condition, thereby establishing that the policy is MSM.

We now show that Conditions 1 and 2 in the Lemma are both satisfied in every time slot, by the activation vector produced by $\pi_{TD}^{(N)}$.

1) When $j_{2m} - j_{2m-1}$ is even, i.e., we have an *odd* length run of nonempty queues: By definition of the indices, this means that Queue $(j_{2m-1}) - 1$ is empty since a run of nonempty queues begins with j_{2m-1}, which from Condition 3 in the definition of $\pi_{TD}^{(N)}$ ensures that $s_{j_{2m-1}-1}(t) = 0$, which means that $s_{j_{2m-1}-1}(t) = 1$ from Condition 1. Thereafter, since all queues between Queues j_{2m-1} and j_{2m} (including these two) are nonempty, the policy alternates between Conditions 1 and 2, scheduling every alternate queue and thus satisfying Condition 1 in Lem. 19.
2) When $j_{2m} - j_{2m-1}$ is odd, i.e., we have an *even* length run of nonempty queues: Once again $\pi_{TD}^{(N)}$ schedules every alternate queue within this run starting with Queue j_{2m-1}, and in the process, satisfies Condition 2a in Lem. 19.

Since this holds true in every slot, the policy is MSM. ■

J. Proof of Thm. 9

The policy $\pi_{SP}^{(2N-1)}$ is formed by splicing together $\pi_{TD}^{(N)}$ and $\pi_{BU}^{(N)}$. This means that $\pi_{SP}^{(2N-1)}$ restricted to Queues 1 to N is $\pi_{BU}^{(N)}$ and restricted to Queues N to $2N-1$ is $\pi_{TD}^{(N)}$. The throughput-optimality of $\pi_{TD}^{(N)}$ and $\pi_{BU}^{(N)}$ means that for every $\boldsymbol{\lambda} \in \Lambda_N$,

$$\limsup_{T\to\infty} \frac{1}{T} \sum_{t=0}^{t-1} \sum_{i=N}^{2N-1} \mathbb{E}_{\pi_{TD}^{(N)}} Q_i(t) < \infty, \text{ and} \tag{32}$$

$$\limsup_{T\to\infty} \frac{1}{T} \sum_{t=0}^{t-1} \sum_{i=1}^{N} \mathbb{E}_{\pi_{BU}^{(N)}} Q_i(t) < \infty. \tag{33}$$

Notice in particular the indices of the inner summations in the above inequalities. Now, for every $\boldsymbol{\lambda} \in \Lambda_{2N-1}$ define $\boldsymbol{\lambda}_{1:N} = [\lambda_1, \cdots, \lambda_N]$ and $\boldsymbol{\lambda}_{N:2N-1} = [\lambda_N, \cdots, \lambda_{2N-1}]$ and notice that by the definition of Λ_N, $\boldsymbol{\lambda}_{1:N} \in \Lambda_N$ and $\boldsymbol{\lambda}_{N:2N-1} \in \Lambda_N$, which means that (32) and (33) are still separately true. The proof concludes when we observe that

$$\begin{aligned} & \limsup_{T\to\infty} \frac{1}{T} \sum_{t=0}^{t-1} \sum_{i=1}^{2N-1} \mathbb{E}_{\pi_{SP}^{(2N-1)}} Q_i(t) \\ \le & \limsup_{T\to\infty} \frac{1}{T} \sum_{t=0}^{t-1} \sum_{i=N}^{2N-1} \mathbb{E}_{\pi_{TD}^{(N)}} Q_i(t) \\ & + \limsup_{T\to\infty} \frac{1}{T} \sum_{t=0}^{t-1} \sum_{i=1}^{N} \mathbb{E}_{\pi_{BU}^{(N)}} Q_i(t) \\ < & \infty \end{aligned}$$

■

K. Proof of Lem. 8

We know that $\pi_{TD}^{(N)}$ and $\pi_{BU}^{(N)}$ they produce a single activation vector $\mathbf{s}(t) \in \{0,1\}^N$ for every $\boldsymbol{\zeta}(t) \in \{0,1\}^N$. We have also seen that $\pi_{SP}^{(2N-1)}$ induces $\pi_{BU}^{(N)}$ on Queues 1 through N, and $\pi_{TD}^{(N)}$ on Queues N through $2N-1$. Given any occupancy vector for the new system $\boldsymbol{\zeta}(t) \in \{0,1\}^{2N-1}$ notice that $\pi_{BU}^{(N)}$ maps coordinates 1 through N to a single activation vector and $\pi_{TD}^{(N)}$ maps coordinates N through $2N-1$ to a single activation vector, with a non-conflicting overlap at Queue N. Thus, every $\boldsymbol{\zeta}(t) \in \{0,1\}^{2N-1}$ gets mapped to a unique activation vector $\boldsymbol{s}(t) \in \{0,1\}^{2N-1}$, resulting in an admissible policy. ■

L. Proof of Prop. 10

Before we prove the stability of $\pi_{TD}^{(4)}$, we will need the following two lemmas.

Lemma 20. Let A and B be two independent random variables, with A taking values in $\{0,1,2,\cdots\}$ and B taking values in $\{0,1\}$. Define $Z := (A-B)^+$. Then,

$$\mathbb{E}Z = \mathbb{E}A - P\{B=1\}(1-P\{A=0\}). \tag{34}$$

PROOF. Let $p_k = P\{A=k\}, k \geq 0$, and $q = P\{B=1\}$. Then, $P\{Z=0\} = P\{A=0\} + P\{A=1,B=1\} = p_0 + p_1 q$, and for all $k \geq 1$,

$$\begin{aligned} P\{Z=k\} &= P\{A=k,B=0\} + P\{A=k+1,B=1\} \\ &= p_k(1-q) + p_{k+1}q. \end{aligned} \tag{35}$$

Hence,

$$\begin{aligned} \mathbb{E}Z &= \sum_{k=1}^{\infty} kP\{Z=k\} \\ &= \sum_{k=1}^{\infty} k\left(p_k(1-q)+p_{k+1}q\right) \\ &= (1-q)\sum_{k=1}^{\infty} kp_k + q\sum_{k=1}^{\infty}(k+1\ -1)p_{k+1} \\ &= (1-q)\mathbb{E}A + q\sum_{k=1}^{\infty}(k+1)p_{k+1} - q\sum_{k=1}^{\infty}p_{k+1} \\ &= (1-q)\mathbb{E}A + q(\mathbb{E}A - p_1) - q(1-p_1-p_0) \\ &= \mathbb{E}A - q(1-p_0). \end{aligned} \tag{36}$$

■

Lemma 21. Under the policy $\pi_{TD}^{(3)}$,

$$\lim_{t\to\infty} P\{Q_1(t)=0,Q_2(t)=0,Q_3(t)=0\} = (1-\lambda_1)\left(1-\frac{\lambda_2}{1-\lambda_1}\right)\left(1-\frac{\lambda_3}{1-\lambda_2}\right). \tag{37}$$

PROOF. We have already shown that the policy $\pi_{TD}^{(3)}$ is throughput-optimal. From this we see that

- the vector-valued process $\{[Q_1(t),Q_2(t),Q_3(t)], t \geq 0\}$ is strongly stable, under $\pi_{TD}^{(3)}$ and hence, also a positive recurrent DTMC.
- Recall that while $s_i(t)$ is used to indicate if service is "offered" to Queue i at time t, $D_i(t)$ indicates if a packet actually leaves Queue i at the end of that slot, i.e., $D_i(t) = s_i(t)\mathbb{I}_{\{Q_i(t)>0\}}$. The proof of throughput-optimality of $\pi_{TD}^{(3)}$ (Thm. 2) already showed that when Queues 1 and 2 are started out in their steady state distributions,

$$P\{s_3(t)=1\} = 1-\lambda_2. \tag{38}$$

So, assume queues 1, 2 and 3 are started out in their steady state distributions. Since $Q_3(t+1) = (Q_3(t)-s_3(t))^+ + A_3(t+1)$, using the fact that Queue 3 is in steady state and Lem. 20, we see that

$$\begin{aligned} \mathbb{E}Q_3(t+1) &= \mathbb{E}Q_3(t) \\ &\quad -P\{s_3(t)=1\}(1-P\{Q_3(t)=0\}) + \lambda_3, \\ \Rightarrow 1-P\{Q_3(t)=0\} &= \frac{\lambda_3}{P\{s_3(t)=1\}}, \\ \Rightarrow P\{Q_3(t)=0\} &= 1-\frac{\lambda_3}{1-\lambda_2}, \text{ in the } \textit{steady state}. \end{aligned} \tag{39}$$

Next, note that under $\pi_{TD}^{(3)}$, the system[17] transmits a packet whenever it is nonempty, i.e., it never so happens that the system is nonempty in a slot and none of the queues is served in that slot. Secondly, the system transmits *two* packets in a slot iff both queues 1 and 3 are nonempty.

Now, define, for all $t \geq 0$, $Q(t) := Q_1(t)+Q_2(t)+Q_3(t)$, and $A(t) := A_1(t)+A_2(t)+A_3(t)$. Then, following the above argument,

$$Q(t+1) = Q(t) - \mathbb{I}_{\{Q(t)>0\}} - \mathbb{I}_{\{Q_1(t)>0,Q_3(t)>0\}} + A(t+1). \tag{40}$$

Note that the mean arrival rate to the three queue subsystem is $\mathbb{E}A(t) = \lambda_1+\lambda_2+\lambda_3$. Taking expectation on both sides of the above equation and letting $t \to \infty$, we get

$$\begin{aligned} \lim_{t\to\infty}\mathbb{E}Q(t+1) &= \lim_{t\to\infty}\mathbb{E}Q(t) - \lim_{t\to\infty}P\{Q(t)>0\} \\ &\quad -\lim_{t\to\infty}P\{Q_1(t)>0,Q_3(t)>0\} \\ &\quad +\lambda_1+\lambda_2+\lambda_3. \\ \Rightarrow \lim_{t\to\infty}P\{Q(t)>0\} &= -\lim_{t\to\infty}P\{Q_1(t)>0,Q_3(t)>0\} \\ &\quad +\lambda_1+\lambda_2+\lambda_3. \\ &\overset{\dagger}{=} -\lim_{t\to\infty}P\{A_1(t)=1\}P\{Q_3(t)>0\} \\ &\quad +\lambda_1+\lambda_2+\lambda_3 \\ &= -\frac{\lambda_1\lambda_3}{1-\lambda_2}+\lambda_1+\lambda_2+\lambda_3 \\ &= 1-\left((1-\lambda_1)\left(1-\frac{\lambda_2}{1-\lambda_1}\right)\right. \\ &\quad \left.\times\left(1-\frac{\lambda_3}{1-\lambda_2}\right)\right). \end{aligned}$$

■

We now continue with the proof of Prop. 10. From the definition of $\pi_{TD}^{(4)}$, we see that Queue 4 is offered service under one of the following conditions.

- Queue 1 is nonempty and Queue 3 is empty
- Queue 1 is empty and Queue 2 is nonempty
- Queues 1, 2 and 3 are all empty.

[17] In this proof, by "system," we mean the 3 queues system.

Let us now compute the probability that Queue 4 is offered service in a slot, under the assumption that queues 1, 2 and 3 are started out in stationarity.

$$\begin{aligned} P\{s_4(t)=1\} &= P\{Q_1(t)>0, Q_3=0\} + P\{Q_1(t)=0, Q_2(t)>0\} \\ &+ P\{Q_1(t)=0, Q_2(t)=0, Q_3(t)=0\} \\ &\stackrel{\star}{=} \lambda_1\left(1-\frac{\lambda_3}{1-\lambda_2}\right) + \left((1-\lambda_1)\frac{\lambda_3}{1-\lambda_2}\right) \\ &+ \left((1-\lambda_1)\left(1-\frac{\lambda_2}{1-\lambda_1}\right)\left(1-\frac{\lambda_3}{1-\lambda_2}\right)\right) \\ &= 1-\lambda_3 \\ &> \lambda_4 \end{aligned}$$

In equality $\star$, we used the result of Lem. 21. Now, using the same drift argument as in the proof of throughput-optimality of $\pi_{TD}^{(3)}$ on $Q_4(t)$, we see that the policy is throughput-optimal ■

M. Analyzing the priority policies in greater detail

We will now make a few more observations about $\pi_{TD}^{(3)}$ and $\pi_{TD}^{(4)}$. In what follows, we will drop the time index and represent $Q_i(t)$ by Q_i to simplify notation.

$$\begin{aligned} P\{Q_3>0, Q_1=0, Q_2>0\} &= P\{Q_3>0, Q_2>0\} \\ &\times P\{Q_1=0 | Q_3>0, Q_2>0\} \\ &= (1-\lambda_1)P\{Q_3>0, Q_2>0\} \end{aligned} \tag{41}$$

Next, recall that under $\pi_{TD}^{(4)}$, Queue 4 is *offered* service (i.e., $s_4=1$) whenever either Queue 3 is empty, or when Queue 3 is nonempty, but Queue 1 is empty *and* Queue 2 is non empty. Additionally, from Eqn. 41 we gather that $P\{s_4(t)=1\} = 1-\lambda_3$. Hence,

$$\begin{aligned} P\{s_4(t)=1\} &= 1-\lambda_3 \\ &= P\{Q_3=0\} \\ &+ P\{Q_3>0, Q_1=0, Q_2>0\} \\ &\stackrel{\dagger a}{=} \left(1-\frac{\lambda_3}{1-\lambda_2}\right) \\ &+ (1-\lambda_1)P\{Q_3>0, Q_2>0\} \\ \Rightarrow P\{Q_3>0, Q_2>0\} &= \frac{1}{1-\lambda_1}\left(1-\lambda_3-\left(1-\frac{\lambda_3}{1-\lambda_2}\right)\right) \\ &= \frac{1}{1-\lambda_1}\lambda_3\left(\frac{1}{1-\lambda_2}-1\right) \\ &= \frac{\lambda_2}{1-\lambda_1}\cdot\frac{\lambda_3}{1-\lambda_2} \\ &= P\{Q_2>0\}\cdot P\{Q_3>0\}, \end{aligned} \tag{42}$$

where equality $\dagger a$ above comes from Eqn. (39). Next,

$$\begin{aligned} (1-\lambda_1)\left(1-\frac{\lambda_2}{1-\lambda_2}\right) &= P\{Q_1=0, Q_2=0\} \\ &= P\{Q_1=0, Q_2=0, Q_3>0\} \\ &+ P\{Q_1=0, Q_2=0, Q_3=0\} \\ &\stackrel{\dagger b}{=} (1-\lambda_1)P\{Q_2=0, Q_3>0\} \\ &+ (1-\lambda_1)\left(1-\frac{\lambda_2}{1-\lambda_1}\right)\left(1-\frac{\lambda_3}{1-\lambda_2}\right) \\ \Rightarrow P\{Q_2=0, Q_3>0\} &= \left(1-\frac{\lambda_2}{1-\lambda_1}\right)\frac{\lambda_3}{1-\lambda_2} \\ &= P\{Q_2=0\}\cdot P\{Q_3>0\}, \end{aligned} \tag{43}$$

where, in equality $\dagger b$ we have made use of Lem. 21. Next,

$$\begin{aligned} P\{Q_2=0, Q_3=0\} &= P\{Q_1>0, Q_2=0, Q_3=0\} \\ &+ P\{Q_1=0, Q_2=0, Q_3=0\} \\ &= \lambda_1 P\{Q_2=0, Q_3=0\} \\ &+ (1-\lambda_1)\left(1-\frac{\lambda_2}{1-\lambda_1}\right)\left(1-\frac{\lambda_3}{1-\lambda_2}\right), \end{aligned} \tag{44}$$

which means that

$$\begin{aligned} (1-\lambda_1)P\{Q_2=0, Q_3=0\} &= (1-\lambda_1)\left(1-\frac{\lambda_2}{1-\lambda_1}\right)\left(1-\frac{\lambda_3}{1-\lambda_2}\right) \\ \Rightarrow P\{Q_2=0, Q_3=0\} &= \left(1-\frac{\lambda_2}{1-\lambda_1}\right)\left(1-\frac{\lambda_3}{1-\lambda_2}\right) \\ &= P\{Q_2=0\}\cdot P\{Q_3=0\}. \end{aligned} \tag{45}$$

Finally,

$$\begin{aligned} P\{Q_2>0, Q_3=0\} &= 1-(P\{Q_2>0, Q_3>0\} \\ &+ P\{Q_2=0, Q_3>0\}) + P\{Q_2=0, Q_3=0\} \\ &= 1-\left(\frac{\lambda_2}{1-\lambda_1}\cdot\frac{\lambda_3}{1-\lambda_2} + \left(1-\frac{\lambda_2}{1-\lambda_1}\right)\cdot\frac{\lambda_3}{1-\lambda_2}\right. \\ &+ \left.\left(1-\frac{\lambda_2}{1-\lambda_1}\right)\left(1-\frac{\lambda_3}{1-\lambda_2}\right)\right) \\ &= 1-\left(\frac{\lambda_3}{1-\lambda_2}+1-\frac{\lambda_2}{1-\lambda_1}-\frac{\lambda_3}{1-\lambda_2}\right. \\ &+ \left.\frac{\lambda_2}{1-\lambda_1}\frac{\lambda_3}{1-\lambda_2}\right) \\ &= \frac{\lambda_2}{1-\lambda_1}\left(1-\frac{\lambda_2}{1-\lambda_1}\right) \\ &= P\{Q_2>0\}\cdot P\{Q_3=0\}. \end{aligned} \tag{46}$$

From Eqn. (42), Eqn. (43), Eqn. (45) and Eqn. (46) and we see that the random variables $\mathbb{I}_{\{Q_1>0\}}$, $\mathbb{I}_{\{Q_2>0\}}$ and $\mathbb{I}_{\{Q_3>0\}}$ are **independent**, for every $t \geq 0$ under the condition that the initial queue length vector $[Q_1(0), Q_2(0), Q_3(0)]$ follows the steady state distribution, which always exists since $\pi_{TD}^{(3)}$ is throughput-optimal.

Remark. Since the top-down priority policies $\pi_{TD}^{(3+k)}$, for all $k \geq 0$ induce $\pi_{TD}^{(3)}$ on queues 1, 2 and 3, this independence is *always true* in steady state.

N. Proof of Prop. 11

Recall that under the top-down policies, Queue 1 receives highest priority and is served whenever it is non empty, followed by Queue 2 and so on. $\pi_{TD}^{(5)}$ offers service to Queue 5 (i.e., $s_5(t)=1$) **iff** the following conditions are satisfied

- $Q_1 > 0$ *and* $Q_3 > 0$
- $Q_1 > 0,\ Q_3 = 0,$ *and* $Q_4 = 0,$
- $Q_1 = 0,\ Q_2 > 0,$ *and* $Q_4 = 0,$
- $Q_1 = 0,\ Q_2 = 0,$ *and* $Q_3 > 0$ *and*

- $Q_1 = 0,\ Q_2 = 0,\ Q_3 = 0$ *and* $Q_4 = 0$.

So,

$$\begin{aligned} P\{s_5(t)=1\} &= P\{Q_1>0,\ Q_3>0\} \\ &\quad +P\{Q_1=0,\ Q_2=0,\ Q_3>0\} \\ &\quad +P\{Q_1>0,\ Q_3=0,\ Q_4=0\} \\ &\quad +P\{Q_1=0,\ Q_2>0,\ Q_4=0\} \\ &\quad +P\{Q_1=0,\ Q_2=0,\ Q_3=0,\ Q_4=0\} \\ &\stackrel{\dagger c}{=} \lambda_1\frac{\lambda_3}{1-\lambda_2}+(1-\lambda_1)\left(1-\frac{\lambda_2}{1-\lambda_1}\right)\frac{\lambda_3}{1-\lambda_2} \\ &\quad +P\{Q_1>0,\ Q_3=0,\ Q_4=0\} \\ &\quad +P\{Q_1=0,\ Q_2>0,\ Q_4=0\} \\ &\quad +P\{Q_1=0,\ Q_2=0,\ Q_3=0,\ Q_4=0\}, \end{aligned} \tag{47}$$

where in equality c we have used the independence results of Sec.XI-M. Now, consider the subsystem comprising queues 1, 2 and 4 under this policy and call this subsystem Q_{124}. Since $\pi^{(5)}_{TD}$ restricted to the first 4 queues is essentially $\pi^{(4)}_{TD}$, the top-down policy for the 4 queue system, and since Thm. 10 already showed that $\pi^{(4)}_{TD}$ is throughput-optimal, Q_{124} is a stable subsystem and *has a steady state distribution* which is simply a *marginal* of the distribution of the 4 queue system with Queue 3's coordinate summed out.

The arrival rate to Q_{124} is $\lambda_1+\lambda_2+\lambda_4$. Also, under $\pi^{(4)}_{TD}$ and hence $\pi^{(5)}_{TD}$, the Q_{124} transmits

- At least 1 packet in slots with
  - $Q_1 > 0$, or
  - $Q_1 = 0,\ Q_2 > 0$, or
  - $Q_1 = 0,\ Q_2 = 0,\ Q_3 = 0$ and $Q_4 > 0$, and
- 2 packets in slots with
  - $Q_1 > 0,\ Q_3 = 0,$ and $Q_4 > 0$, or
  - $Q_1 = 0,\ Q_2 > 0,$ and $Q_4 > 0$.

Assume Q_{124} is started out in its steady state. Let $Q(t) = \sum_{i\in Q_{124}} Q_i(t)$, and $A(t) = \sum_{i\in Q_{124}} A_i(t)$, for all $t \geq 0$. Consequently,

$$\begin{aligned} Q(t+1) &= Q(t) - \mathbb{I}_{\{Q_1(t)>0\}} - \mathbb{I}_{\{Q_1(t)=0,Q_2(t)>0\}} \\ &\quad -\mathbb{I}_{\{Q_1(t)=0,Q_2(t)=0,Q_3(t)=0,Q_4(t)>0\}} \\ &\quad -\mathbb{I}_{\{Q_1(t)>0,Q_3(t)=0,Q_4(t)>0\}} \\ &\quad -\mathbb{I}_{\{Q_1(t)=0,Q_2(t)>0,Q_4(t)>0\}} + A(t+1) \\ \mathbb{E}Q(t+1) - \mathbb{E}Q(t) &= -P\{Q_1(t)>0\} \\ &\quad -P\{Q_1(t)=0,Q_2(t)>0\} \\ &\quad -P\{Q_1(t)=0,Q_2(t)=0, \\ &\quad Q_3(t)=0,Q_4(t)>0\} \\ &\quad -P\{Q_1(t)>0,Q_3(t)=0,Q_4(t)>0\} \\ &\quad -P\{Q_1(t)=0,Q_2(t)>0,Q_4(t)>0\} \\ &\quad +\mathbb{E}A(t+1), \end{aligned}$$

which, using the fact that $\mathbb{E}Q(t+1) - \mathbb{E}Q(t) = 0$ in the steady state and that $\mathbb{E}A(t+1) = \lambda_1+\lambda_2+\lambda_4$, gives us

$$\begin{aligned} &P\{Q_1(t)=0,Q_2(t)=0,Q_3(t)=0,Q_4(t)>0\} \\ &+P\{Q_1(t)>0,Q_3(t)=0,Q_4(t)>0\} \\ &+P\{Q_1(t)=0,Q_2(t)>0,Q_4(t)>0\} \\ &= \lambda_1+\lambda_2+\lambda_4 - P\{Q_1(t)>0\} - P\{Q_1(t)=0,Q_2(t)>0\} \\ &= \lambda_1+\lambda_2+\lambda_4-\lambda_1-(1-\lambda_1)\frac{\lambda_2}{1-\lambda_1} \\ &= \lambda_4. \end{aligned} \tag{48}$$

Notice that

$$\begin{aligned} &P\{Q_1(t)=0,Q_2(t)=0,Q_3(t)=0,Q_4(t)>0\} \\ &+P\{Q_1(t)>0,Q_3(t)=0,Q_4(t)>0\} \\ &+P\{Q_1(t)=0,Q_2(t)>0,Q_4(t)>0\} \\ &+P\{Q_1(t)=0,Q_2(t)=0,Q_3(t)=0,Q_4(t)=0\} \\ &+P\{Q_1(t)>0,Q_3(t)=0,Q_4(t)=0\} \\ &+P\{Q_1(t)=0,Q_2(t)>0,Q_4(t)=0\} \\ &= P\{Q_1(t)=0,Q_2(t)=0,Q_3(t)=0\} \\ &+P\{Q_1(t)>0,Q_3(t)=0\} \\ &+P\{Q_1(t)=0,Q_2(t)>0\}. \end{aligned}$$

Using Eqn. 48 on the first three terms on the LHS of the above equation and invoking the independence results in Sec. XI-M on the RHS, we get

$$\begin{aligned} &\lambda_4 + P\{Q_1(t)=0,Q_2(t)=0,Q_3(t)=0,Q_4(t)=0\} \\ &+P\{Q_1(t)>0,Q_3(t)=0,Q_4(t)=0\} \\ &+P\{Q_1(t)=0,Q_2(t)>0,Q_4(t)=0\} \\ &= (1-\lambda_1)\left(1-\frac{\lambda_2}{1-\lambda_1}\right)\left(1-\frac{\lambda_3}{1-\lambda_2}\right)+\lambda_1\left(1-\frac{\lambda_3}{1-\lambda_2}\right) \\ &+(1-\lambda_1)\frac{\lambda_2}{1-\lambda_1}, \end{aligned}$$

which means that,

$$\begin{aligned} &P\{Q_1(t)=0,Q_2(t)=0,Q_3(t)=0,Q_4(t)=0\} \\ &+P\{Q_1(t)>0,Q_3(t)=0,Q_4(t)=0\} \\ &+P\{Q_1(t)=0,Q_2(t)>0,Q_4(t)=0\} \\ &= (1-\lambda_1)\left(1-\frac{\lambda_2}{1-\lambda_1}\right)\left(1-\frac{\lambda_3}{1-\lambda_2}\right)+\lambda_1\left(1-\frac{\lambda_3}{1-\lambda_2}\right) \\ &+(1-\lambda_1)\frac{\lambda_2}{1-\lambda_1}-\lambda_4. \end{aligned}$$

Substituting this on the RHS of Eqn. 47, we get

$$\begin{aligned} P\{S_5(t)=1\} &= \lambda_1\frac{\lambda_3}{1-\lambda_2}+(1-\lambda_1)\left(1-\frac{\lambda_2}{1-\lambda_1}\right)\frac{\lambda_3}{1-\lambda_2} \\ &+ \lambda_1\left(1-\frac{\lambda_3}{1-\lambda_2}\right)+(1-\lambda_1)\left(1-\frac{\lambda_2}{1-\lambda_1}\right)\left(1-\frac{\lambda_3}{1-\lambda_2}\right) \\ &+ (1-\lambda_1)\frac{\lambda_2}{1-\lambda_1}-\lambda_4 \\ &= \lambda_1+(1-\lambda_1)\left(1-\frac{\lambda_2}{1-\lambda_1}\right)+(1-\lambda_1)\frac{\lambda_2}{1-\lambda_1}-\lambda_4 \\ &= 1-\lambda_4 \\ &> \lambda_5. \end{aligned}$$

Thus, both $\pi^{(5)}_{TD}$ and $\pi^{(5)}_{BU}$, the top-down and bottom-up policies are stable. ■

O. Proof of Prop. 13

We extend our initial idea of Property $\mathcal{P}$ to prove this proposition as follows. For all $1 \le m \le N$, define $Q_m(t) := \sum_{j\in C_m} Q_{m,j}(t)$, and $D_m(t) := \sum_{j\in C_m} D_{m,j}(t)$. As in the proof of sufficiency of Property $\mathcal{P}$, the idea is to prove that $\tilde{\phi}$ satisfies the following version of the property, which immediately leads to strong stability. For all $2 \le m \le N$,

$$D_1(t) + D_m(t) = 0 \iff Q_1(t) + Q_m(t) = 0, \ \forall t \ge 0. \quad (49)$$

Let $m \ge 2$ in the sequel. By the definition of the departure processes $\{D_i(t), i \ge 1\}$, in every slot $t \ge 0$, $Q_1(t) + Q_m(t) = 0$ always means $D_1(t) + D_m(t) = 0$. To show the converse, we consider several cases

- $Q_1(t) > 0$ and $Q_m(t) > 0$ means that in one of the 3 steps of the definition of $\tilde{\phi}$, one of the queues in either of these cliques will get scheduled and either $D_1(t) = 1$ or $D_m(t) = 1$.
- $Q_1(t) > 0$ and $Q_m(t) = 0$ means that $\tilde{\phi}$ schedules a nonempty queue in C_1 in step 2, and $D_1(t) = 1$.
- $Q_1(t) = 0$ and $Q_m(t) > 0$ means that $\tilde{\phi}$ schedules a nonempty queue in C_m in either step 1 or step 3, depending on whether the other cliques have nonempty queues. In either case, $D_m(t) = 1$.

The proof of sufficiency of Property $\mathcal{P}$ can now be extended using the Lyapunov function defined below to show that $\tilde{\phi}$ is indeed throughput-optimal.

$$L(\mathbf{Q}(t)) := \sum_{m=2}^{N} (Q_1(t) + Q_m(t))^2 \quad (50)$$

■

P. Proof of Prop. 17

The policy $\theta^{(5L)}_{SP}$ is formed by splicing together $\theta^{(3L)}_{TD}$ and $\theta^{(3L)}_{BU}$. This means that $\theta^{(5L)}_{SP}$ restricted to Queues 1 to 3 is $\theta^{(3L)}_{BU}$ and restricted to Queues 3 to 5 is $\theta^{(3L)}_{TD}$. The throughput-optimality of $\theta^{(3L)}_{TD}$ and $\theta^{(3L)}_{BU}$ means that for every $\boldsymbol{\lambda} \in \Lambda^{(5)}_l$ (defined in (7)),

$$\limsup_{T\to\infty} \frac{1}{T} \sum_{t=0}^{t-1} \sum_{i=3}^{5} \sum_{j=1}^{N_i} \mathbb{E}_{\theta^{(3L)}_{TD}} Q_{i,j}(t) < \infty, \text{ and} \quad (51)$$

$$\limsup_{T\to\infty} \frac{1}{T} \sum_{t=0}^{t-1} \sum_{i=1}^{3} \sum_{j=1}^{N_i} \mathbb{E}_{\theta^{(3L)}_{BU}} Q_{i,j}(t) < \infty. \quad (52)$$

Now, as in Thm. 9, for every $\boldsymbol{\lambda} \in \Lambda^{(5)}_l$ we define two new arrival rate vectors $\boldsymbol{\lambda}_{1:3} = [\lambda_1, \lambda_2, \lambda_3]$ and $\boldsymbol{\lambda}_{3:5} = [\lambda_3, \lambda_4, \lambda_5]$ and note that by the definition of the set $\Lambda^{(N)}_l$, $\boldsymbol{\lambda}_{1:3}$ and $\boldsymbol{\lambda}_{3:5}$ are both in $\Lambda^{(3)}_l$, which means that (51) and (52) are still separately true. We conclude the proof by observing that

$$\begin{aligned} & \limsup_{T\to\infty} \frac{1}{T} \sum_{t=0}^{t-1} \sum_{i=1}^{5} \sum_{j=1}^{N_i} \mathbb{E}_{\phi^{(5)}_{SP}} Q_{i,j}(t) \\ \le & \limsup_{T\to\infty} \frac{1}{T} \sum_{t=0}^{t-1} \sum_{i=3}^{5} \sum_{j=1}^{N_i} \mathbb{E}_{\theta^{(3L)}_{TD}} Q_{i,j}(t) \\ & + \limsup_{T\to\infty} \frac{1}{T} \sum_{t=0}^{t-1} \sum_{i=1}^{3} \sum_{j=1}^{N_i} \mathbb{E}_{\theta^{(3L)}_{BU}} Q_{i,j}(t) \\ < & \infty \end{aligned}$$

■

Q. Throughput-optimality of $\Phi^{(S)}_{IC}(T)$

The proof consists of two parts. We will first prove that $\Phi^{(S)}_{IC}(T)$ specialized to a single collocated network, i.e., a single clique is throughput-optimal and then use a new version of property $\mathcal{P}$ to complete the proof for our "star-of-cliques" interference graphs (of the type shown in Fig. 2b).

Suppose the system only knows $\boldsymbol{\zeta}(t) \in \{0,1\}^N$, at times $t = 0, T, 2T, \cdots$. Arrivals in the k^{th} frame are *not* served in the k^{th} frame. We denote by ψ_T, the scheduling policy that, during $kT, kT+1, \ldots, KT+T-1$, serves every queue known to be nonempty at kT until either

1) The next frame, i.e., $k+1$ begins, or
2) All packets queued in the system until the beginning of slot kT have been served. In this case the system obviously idles until the next frame begins.

Since only one queue can be served in any slot, the capacity region of this system is $\left\{\boldsymbol{\lambda} \in \mathbb{R}^N_+ \mid \sum_{i=1}^N \lambda_i < 1\right\}$. In what follows, we will analyze the process $\{\mathbf{q}(k), k \ge 0\}$, where $\mathbf{q}(k) := \mathbf{Q}(kT)$.

Lemma 22. Under ψ_T, for any $\boldsymbol{\lambda}$ inside the capacity region,

- the process $\{\mathbf{q}(k), k \ge 0, \}$ is strongly stable, i.e., ψ_T is throughput-optimal, and
- mean packet delay under ψ_T is *linear* in T which means that there exists an $\alpha \in \mathbb{R}_+$, such that

$$\mathbb{E}_{\psi_T} \sum_{i=1}^{N} Q_i(kT) \le \alpha T, \ \forall k \ge 0. \quad (53)$$

PROOF. Let $A_i[x, y]$ denote the number of arrivals to Queue i over the slots $x, x+1, \ldots, y$. Since the arrivals are all Bernoulli, $A_i[x, y]$ is a Binomial$(y - x + 1, \Lambda_i)$ random variable. Denote the total system backlog at kT by $q(k) := \sum_{i=1}^N q_i(k)$ and total arrival to the system during the k^{th} frame by $A(k+1) := \sum_{i=1}^N A_i[kT+1, k(T+1)]$. It is then easy to see that

$$q(k+1) = (q(k) - T)^+ + A(k+1), \quad (54)$$

since ψ_T serves the network until all $q(k)$ packets leave, if $q(k) < T$, or exactly T packets depart (this happens when the

k^{th} frame begins with at least T packets in the network). With this we get,

$$\begin{aligned}\mathbb{E}_{\psi_T}\left[q^2(k+1)-q^2(k)\mid q(k)=q\right] &\leq q^2+(N^2+1)T^2\\ &- 2qT\left(1-\sum_{i=1}^{N}\Lambda_i\right),\\ &= q^2+(N^2+1)T^2-2\epsilon q,\end{aligned}$$

where $\epsilon := \left(1-\sum_{i=1}^{N}\Lambda_i\right) > 0$. Taking expectations on both sides of the above equation, we get

$$\begin{aligned}\mathbb{E}_{\psi_T}\left[q^2(k+1)-q^2(k)\right] &\leq \mathbb{E}_{\psi_T}q^2(k)+(N^2+1)T\\ &- 2\epsilon\mathbb{E}_{\psi_T}q(k),\\ \mathbb{E}_{\psi_T}\sum_{i=1}^{N}Q_i(kT) &\overset{\star}{\leq} \frac{(N^2+1)}{2\epsilon}T, \qquad (55)\\ \Rightarrow \limsup_{k\to\infty}\frac{1}{kT}\sum_{l=0}^{k-1}\mathbb{E}_{\psi_T}\sum_{i=1}^{N}Q_i(lT) &< \infty, \qquad (56)\end{aligned}$$

In inequality $\star$, we have used the fact that $\mathbb{E}_{\psi_T}q^2(k+1)\geq 0$. In particular, Eqn. 56 shows that the system is strongly stable under ψ_T and setting $\alpha = \frac{(N^2+1)}{2\epsilon}$, and using Little's theorem along with Eqn. (55) we see that mean packet delays are linear in T. ■

The proof of throughput-optimality of $\Phi^{(T)}$ follows by using the above lemma with the $(N-1)$ queue lengths $\left[\left(\sum_{j\in C_1}Q_j(t)+\sum_{j\in C_k}Q_j(t)\right),\ 2\leq k\leq N\right]$. It also means that delay with $\Phi^{(T)}$ increases linearly in T. ■

R. Proof of Prop. 15

This proof proceeds along the same lines as the proof of delay optimality of Policy $\tilde{\pi}^{(3)}_{IQ}$ that we presented in Sec. XI-G.$\phi^{(S)}_{IC}$ and $\tilde{\phi}$ differ only when every peripheral clique has a nonempty queue and behave identically otherwise. Verifying the conditions required to establish sample pathwise and hence, stochastic ordering are very similar to our proof of delay-optimality of $\tilde{\pi}^{(3)}_{IQ}$ and will not be repeated. ■

S. Proof of Prop. 16

Let $Q_i(t) := \sum_{j=1}^{N_i}Q_{i,j}(t)$ be the total backlog of Clique i, i.e., C_i, at the beginning of time slot t and let the total arrival rate to C_i be denoted by $\lambda_i := \sum_{j\in C_i}\lambda_{i,j} = \sum_{j=1}^{N_i}\lambda_{i,j}$. Define $Q_{1,2}(t) := Q_1(t)+Q_2(t)$. Notice that Clique 1 is scheduled for service in every slot in which any queue in it has a packet and whenever C_1 is not scheduled, C_2 is scheduled provided it is non empty. So, we have that

$$Q_{1,2}(t+1) = Q_{1,2}(t) - \mathbb{I}_{\{Q_{1,2}(t)>0\}} + A_{1,2}(t+1), \qquad (57)$$

where $A_{1,2}(t+1),\ t\geq 0$ is the total number of arrivals to C_1 and C_2 at the beginning of slot $t+1$, and $\mathbb{E}A_{1,2}(t+1) = \lambda_1+\lambda_2$. It is easy to show that Cliques 1 and 2 are stable under this policy (a simple sum of queue length squares Lyapunov function suffices), which means that there exists a stationary distribution for the process $\{Q_{1,2}(t),\ t\geq 0\}$. Now, from (7) with $N=3$ cliques, we know that $\lambda_1+\lambda_2<1$. Taking expectation on both sides of the equation *in steady state*, we get

$$\begin{aligned}\mathbb{E}Q_{1,2}(t+1) &= \mathbb{E}Q_{1,2}(t) - P\left\{Q_{1,2}(t)>0\right\}+\lambda_1+\lambda_2,\\ \Rightarrow P\left\{Q_{1,2}(t)=0\right\} &= 1-\lambda_1-\lambda_2 \qquad (58)\end{aligned}$$

So, since a non empty queue in C_3 is served in every slot in which either

- there is a non empty queue in C_1, or
- there are no non empty queues in C_1, or C_2.

With this we see that the *offered service process* to Clique 3, i.e., $\{S_{C_3}(t),\ t\geq 0\}$ satisfies

$$\begin{aligned}P\{S_{C_3}(t)=1\} &= \lambda_1+(1-\lambda_1-\lambda_2)\\ &= 1-\lambda_2>\lambda_3. \qquad (59)\end{aligned}$$

Notice that $P\{S_{C_3}(t)=1\}$ is independent of $Q_3(t)$. Hence, repeating the drift argument from Thm. 2 that showed the throughput-optimality of $\pi^{(3)}_{TD}$ on $Q_3(t)$, we see that C_3 is also stable which means that $\theta^{(3L)}_{TD}$ is throughput-optimal. ■

T. Proof of Prop. 14

We first show that $\phi^{(S)}_{IC}$ satisfies Eqn. 49 at every $t\geq 0$. Thereafter, the analysis in the proof of $\tilde{\phi}^{(S)}_{IC}$ using the same Lyapunov function as in Eqn. (50) can be used to establish strong stability. Let $m\geq 2$ in the sequel. By the definition of the departure processes $\{D_i(t), i\geq 1\}$, in every slot $t\geq 0$, $Q_1(t)+Q_m(t)=0$ always means $D_1(t)+D_m(t)=0$. To show the converse, we consider several cases

- $Q_1(t)>0 \Rightarrow D_1(t)=1$.
- $Q_1(t)=0$ and $Q_m(t)>0$ means that $\phi^{(S)}_{IC}$ schedules a nonempty queue in C_m in step 1 or step 2 ensuring $D_m(t)=1$.

The proof now uses the same Lyapunov function as XI-O and proceeds along the same lines. ■

U. Proof of Thm. 18

We begin by first analyzing the service processes to different queues under $\phi^{(S)}_{CS}$. This will yield important insights into how the stability proof should proceed.

a) Service Processes under $\phi^{(S)}_{CS}$ :: For the purposes of this proof, we relabel the queue in the central clique as $Q_c(t)$ and assume that the peripheral cliques are numbered $C_1,\cdots,C_N$. Clearly, since the central clique gets maximum priority and Queue c, the only queue in this clique, is served whenever it is nonempty, it behaves as a $Geo/D/1$ queue with service time equal to 1 slot. We now move on to queues in the peripheral cliques. WLOG we consider clique C_1 and $Q_{1,1}$ in it and note that every clique is running an *exhaustive service policy* locally. Fig. 13 shows a sample path of the queue length-evolution process $Q_{1,1}(t)$. Note that Queue c is served in every slot in which there is an arrival to it and that due to the Bernoulli nature of the arrival processes, the interarrival duration is Geometric with mean $\frac{1}{1-\lambda_c}$. A packet in $Q_{1,1}$ that reaches the Head-of-Line (HOL) position therefore sees a service duration that is Geometric with mean $\frac{1}{1-\lambda_c}$. This, of course, is true for every queue in the peripheral cliques.

1) Glossary II:

* $\mathbf{V}(t)$: The $(\mathcal{N}\times 1)$-dimensional vector containing information about the number of slots since each of the queues in the SoC network over which $\phi_{CS}^{(S)}$ is defined, was last served before time slot t.
* Δ : The length of a server vacation in the proof of throughput-optimality of $\phi_{CS}^{(S)}$, once again, in Sec. XI-U1.
* $I(\cdot)$: index of the queue currently having channel access.
* n : the slot in which Queue $I(n)$ begins transmission.
* $n+1$: the slot in which channel access is granted to the next peripheral queue.
* m_l : instant at which the l^{th} (Vacation + Busy Period) begins.
* $R_{I(n)}$: the number of (Vacation + Busy Period)'s for Queue $I(n)$.
* λ_c : packet arrival rate at the central queue, i.e., Queue c.
* $n_i,\ i\geq 0$: instants at which busy periods of Queue $I(n)$ begin. Clearly, $n_0 = n$.

Once $Q_{1,1}$ becomes empty, the rest of the clique does not necessarily obtain this knowledge in that same slot. It depends on whether the central clique is empty or not. For example, in slot m_0 in Fig. 13, $Q_{1,1}$ has become empty, but Queue c has received an arrival which means that it is Queue c that is served, power is sensed in minislot 1 itself, and the protocol $\phi_{CS}^{(S)}$ never enters Step 2b. The other queues in C_1 don't know if $Q_{1,1}$ is empty and so, in the next slot in which Queue c is empty, it is $Q_{1,1}$ that is given channel access and if, by then it has received any arrivals, it begins another busy period. This is what happens in Fig. 13 until instant m_1 and this entire process repeats resulting in a random number of busy periods of $Q_{1,1}$ until the instant when *both* Queue c *and* $Q_{1,1}$ are found empty. This happens in slot $n_{R_{I(n)}}$ in the figure. At this instant, $\phi_{CS}^{(S)}$ enters Step. 2b and the queue with the largest V_i takes over. Notice that there are several portions labelled "Vacation" in the figure. A *Vacation* is the duration between a peripheral queue becoming empty and the first time since then that it is granted channel access since the central queue is empty. The durations of these vacations are also distributed Geometric with mean $\frac{1}{1-\lambda_c}$. To summarize, the service to a peripheral queue under $\phi_{CS}^{(S)}$ consists of a random number of (Busy Period + Vacation) durations.

The proof will focus on analyzing $\phi_{CS}^{(S)}$ restricted to a single clique and this analysis will be extended later to show that the entire star-of-cliques system is stable. For now, WLOG, we focus on clique C_1. Clearly, the queue length vector process $\mathbf{Q}(t)$ under $\phi_{CS}^{(S)}$ is a DTMC. We prove the throughput-optimality of $\phi_{CS}^{(S)}$ using a drift argument for the queue length vector process $\mathbf{Q}(n)$ which is $\mathbf{Q}(t)$ sampled at instants $n\geq 0$ when a new peripheral queue is granted channel access (see Fig. 13). We define the following quantities

- n : the slot in which Queue $I(n)$ (here $Q_{1,1}$) begins transmission.
- $I(n)$: the queue having channel access
- $n+1$: the slot in which channel access is granted to the next peripheral queue in C_1.
- m_l : instant at which the l^{th} (Vacation + Busy Period) begins.
- $R_{I(n)}$: the number of (Vacation + Busy Period)'s for Queue $I(n)$, here $Q_{1,1}$.
- λ_c : packet arrival rate at Queue c.
- $n_i,\ i\geq 0$: instants at which busy periods of Queue $I(n)$ begin. Clearly, $n_0 = n$.

Note: For the reader's convenience we provide another short glossary (apart from the one in Sec. XI-B) containing the definitions of some of the important quantities used in the proof in Sec. XI-U1 below.

PROOF. Denote by ρ_j the load on Queue j, i.e., $\rho_j = \lambda_j \mathbb{E}B = \lambda_j \frac{1}{1-\lambda_c}$, and by $\rho = \sum_{j=1}^{\mathcal{N}_1}\rho_j$ the total load on the clique. The interference constraints dictate that $\rho < 1$. Let $b = \mathbb{E}B = \frac{1}{\lambda_c}$ Define $\Delta = \inf\{m > m_0 | A_c(m) = 0\} - m_0$, i.e., the duration after m_0 until the slot without any arrival to Queue c (the central queue). Note that the vacation may be of length 0 slots as well, since when the busy period of Queue $I(n)$ ends, Queue c could be empty. Therefore, $\forall k\geq 0\ P\{\Delta = k\} = \lambda_c^k(1-\lambda_c)$, and the mean of Δ is

$$\mathbb{E}\Delta = \sum_{k=0}^{\infty} k\lambda_c^k(1-\lambda_c) = \frac{\lambda_c}{1-\lambda_c} \tag{60}$$

$$\begin{aligned}\mathbb{E}A_j(\Delta) &= \mathbb{E}\left(\mathbb{E}\left[A_j(\Delta)\mid\Delta\right]\right) = \mathbb{E}\left(\lambda_j\mathbb{E}\Delta\right)\\ &= \lambda_c\frac{\lambda_j}{1-\lambda_c} = \lambda_c\rho_j\end{aligned} \tag{61}$$

Now, notice that

$$Q_j(n_1) = \begin{cases} Q_j(n) + \underbrace{A_j\left(G_{I(n)}\left(Q_{I(n)}(n)\right)\right)}_{\text{arrivals to Queue } j \text{ during busy period of Queue } I(n)} \\ \qquad \underbrace{+A_j(\Delta)}_{\text{arrivals during the subsequent vacation}} \quad , \quad \text{for } j\neq I(n) \\ A_j(\Delta), \quad \text{for } j = I(n).\end{cases} \tag{62}$$

Using (61) and (62), we get

$$\begin{aligned} b\mathbb{E}\left[Q_j(n_1)|\mathbf{Q}(n)\right] &\leq bQ_j(n) + bQ_{I(n)}(n)\frac{\rho_j}{1-\rho_{I(n)}} + b\lambda_c\rho_j \\ \mathbb{E}\left[\sum_{j=1}^{\mathcal{N}_1} bQ_j(n_1)|\mathbf{Q}(n)\right] &\leq \sum_{j=1}^{\mathcal{N}_1} bQ_j(n) + \left(\frac{\rho-\rho_j}{1-\rho_{I(n)}} - 1\right)bQ_{I(n)}(n) \\ &\quad + b\lambda_c\rho \\ &\overset{\star}{=} \sum_{j=1}^{\mathcal{N}_1} bQ_j(n) + \underbrace{h_{I(n)}(\rho-1)bQ_{I(n)}(n)}_{\text{strictly negative}} \\ &\quad + b\lambda_c\rho, \end{aligned} \tag{63}$$

where in equality $\star$ we have defined $h_{I(n)} = \frac{1}{1-\rho_{I(n)}}$. Also observe the fact that on the R.H.S of (63), $(\rho-1)$ is *strictly negative,* by capacity constraints. Similarly,

$$\mathbb{E}\left[\sum_{j=1}^{\mathcal{N}_1} bQ_j(n_2)|\mathbf{Q}(n)\right] = \mathbb{E}\left[\underbrace{\mathbb{E}\left[\sum_{j=1}^{\mathcal{N}_1} bQ_j(n_2)\middle|\mathbf{Q}(n_1),\mathbf{Q}(n)\right]}_{\sigma(\mathbf{Q}(n))\subset\sigma(\mathbf{Q}(n_1),\mathbf{Q}(n))}\middle|\mathbf{Q}(n)\right]$$

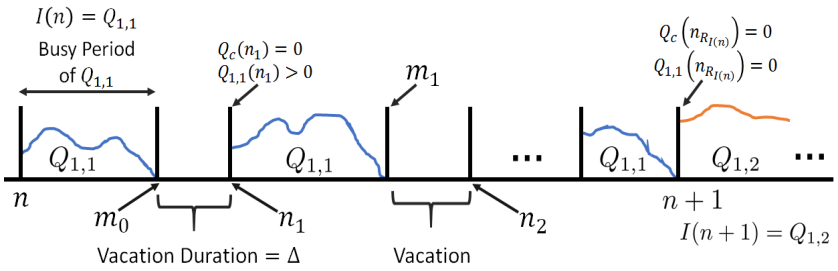


Fig. 13: A sample path illustrating the service process to one of the peripheral queues, here, $Q_{1,1}$. Even with *one* queue in the central clique the resulting service process to peripheral queues is found to be quite complex. Note that here, $I(n) = Q_{1,1}$, *in Clique 1,* while $I(n+1) = Q_{1,2}$ since it was chosen in Step 2b in the definition of $\phi_{CS}^{(S)}$.

$$\begin{aligned}
\mathbb{E}\left[\sum_{j=1}^{\mathcal{N}_1} bQ_j(n_2)|\mathbf{Q}(n)\right] &\overset{\dagger}{=} \mathbb{E}\left[\mathbb{E}\left[\sum_{j=1}^{\mathcal{N}_1} bQ_j(n_2)\Big|\mathbf{Q}(n_1)\right]\Big|\mathbf{Q}(n)\right] \\
&\overset{\star 1}{\leq} \mathbb{E}\left[\sum_{j=1}^{\mathcal{N}_1} bQ_j(n_1)\Big|\mathbf{Q}(n)\right] \\
&\quad + h_{I(n)}(\rho-1)\, b\mathbb{E}\left[Q_{I(n)}(n_1)\Big|\mathbf{Q}(n)\right] + b\lambda_c\rho \\
&= \left(\sum_{j=1}^{\mathcal{N}_1} bQ_j(n)\right. \\
&\quad \left. + h_{I(n)}(\rho-1)\, bQ_{I(n)}(n) + b\lambda_c\rho\right) \\
&\quad + \left(h_{I(n)}(\rho-1)\, b\lambda_c\rho_{I(n)}\right) + b\lambda_c\rho, \\
&= \sum_{j=1}^{\mathcal{N}_1} bQ_j(n) + 2b\lambda_c\rho \\
&\quad + \underbrace{(\rho-1)\, bh_{I(n)}\left(Q_{I(n)} + \rho_{I(n)}\right)}_{\text{strictly negative}}.
\end{aligned}$$

Proceeding similarly,

$$\begin{aligned}
\mathbb{E}\left[\sum_{j=1}^{\mathcal{N}_1} bQ_j(n_k)|\mathbf{Q}(n)\right] &= \sum_{j=1}^{\mathcal{N}_1} bQ_j(n) + kb\lambda_c\rho \\
&\quad + (\rho-1)\, bh_{I(n)}\left(Q_{I(n)}\right. \\
&\quad \left. + (k-1)\rho_{I(n)}\right),\ \forall k \geq 1. \quad (64)
\end{aligned}$$

Equality † follows from the Markovian nature of the evolution of the queue-length vector, and we have used (63) and the fact that $I(n_1) = I(n)$ in inequality $\star 1$. Let us now compute the mean number of secondary busy periods which will inform the choice of k in (64) while computing the conditional drift between instants n and $n+1$. The visit to Queue $I(n)$ ends when it receives 0 arrivals during a vacation. Let the number of vacations during a visit to Queue $I(n)$ be $R_{I(n)}$.

$$\begin{aligned}
P\left(R_{I(n)} = k\right) &= \prod_{l=1}^{k} P\left(A_{I(n)}^{(l)}(\Delta) > 0\right) \\
&\quad \times P\left(A_{I(n)}^{(k+1)}(\Delta) = 0\right),\ \forall k \geq 0. \quad (65)
\end{aligned}$$

But all the $A_{I(n)}^{(l)}(\Delta)$ are iid and

$$\begin{aligned}
P\left(A_{I(n)}^{(l)}(\Delta) = 0\right) &= \sum_{k=0}^{\infty} P\left(A_{I(n)}^{(l)}(\Delta) = 0\Big|\Delta = k\right) \\
&= \sum_{k=0}^{\infty} (1-\lambda_{I(n)})^k \lambda_c (1-\lambda_c)^k \\
&= \frac{1-\lambda_c}{1-(1-\lambda_{I(n)})\lambda_c}, \quad (66)
\end{aligned}$$

Which gives us

$$\mathbb{E}R_{I(n)} = \frac{\lambda_c\lambda_{I(n)}}{1-\lambda_c} + 1 \quad (67)$$

Now, using (67) in (64), we get

$$\begin{aligned}
\mathbb{E}\left[\sum_{j=1}^{\mathcal{N}_1} bQ_j(n_{R_{I(n)}})|\mathbf{Q}(n)\right] &= \sum_{j=1}^{\mathcal{N}_1} bQ_j(n) + \left(\frac{\lambda_c\lambda_{I(n)}}{1-\lambda_c} + 1\right) b\lambda_c\rho \\
&\quad + (\rho-1)\, bh_{I(n)} \\
&\quad \times \left(Q_{I(n)} + \left(\frac{\lambda_c\lambda_{I(n)}}{1-\lambda_c}\right)\rho_{I(n)}\right),
\end{aligned}$$

from which we get the conditional drift as

$$\begin{aligned}
&\mathbb{E}\left[\sum_{j=1}^{\mathcal{N}_1} bQ_j(n+1) - \sum_{j=1}^{\mathcal{N}_1} bQ_j(n)|\mathbf{Q}(n)\right] \\
&= \mathbb{E}\left[\sum_{j=1}^{\mathcal{N}_1} bQ_j(n_{R_{I(n)}}) - \sum_{j=1}^{\mathcal{N}_1} bQ_j(n)|\mathbf{Q}(n)\right] \\
&\leq \left(\frac{\lambda_c\lambda_{I(n)}}{1-\lambda_c} + 1\right) b\lambda_c\rho \\
&\quad + (\rho-1)\, bh_{I(n)}\left(Q_{I(n)} + \left(\frac{\lambda_c\lambda_{I(n)}}{1-\lambda_c}\right)\rho_{I(n)}\right) \\
&< -\epsilon,
\end{aligned}$$

for large enough $Q_{I(n)}$. Invoking the Foster-Lyapunov theorem [37] we see that the chain $\mathbf{Q}(n)$ is positive recurrent. This process can be used repeatedly to show that each of the $\mathcal{N}_1$ DTMCs $\left\{\mathbf{Q}_{n\mathcal{N}_1+K}\right\}_{n=0}^{\infty}$ is positive recurrent for $K = 0, 1, \cdots, \mathcal{N}_1 - 1$. Finally, this same procedure can be repeated for each peripheral clique $C_i, 1 \leq i \leq N$, to show that $\phi_{CS}^{(S)}$ is throughput-optimal.

■

V. Simulation Details

Here we provide the arrival rate vectors for:

- Path graph network simulations in Table II
  1) $N = 4$ queues: $\boldsymbol{\lambda} = [0.49, 0.49, 0.49, 0.49]$
  2) $N = 5$ queues: $\boldsymbol{\lambda} = [0.15, 0.049, 0.95, 0.049, 0.15]$
  3) $N = 15$ queues: $\boldsymbol{\lambda} = [0.80, 0.15, 0.15, 0.15, 0.15, 0.8, 0.049, 0.95, 0.049, 0.8, 0.15, 0.15, 0.15, 0.15, 0.80]$
- Cluster of Cliques simulations in Table III
  1) Star-of-Cliques network: $\boldsymbol{\lambda} = [0.3, 0.3, 0.3, 0.09, 0.9, 0.9]$
  2) Linear-Array-of-Cliques: $\boldsymbol{\lambda} = [0.1, 0.1, 0.1, 0.049, 0.65, 0.3, 0.049, 0.0, 0.0]$