

Throughput Optimal Decentralized Scheduling with Single-bit State Feedback for a Class of Queueing Systems

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, whose main purpose is data collection, we consider the problem of queue scheduling with reduced queue state information. In particular, we consider a time-slotted scheduling model with N sensor nodes, with pair-wise dependence, such that nodes i and $i + 1$, $1 \leq i \leq N - 1$ cannot transmit together. We develop new throughput-optimal scheduling policies requiring only the empty-nonempty state of each queue that we term *Queue Nonemptiness-Based* (QNB) policies. We revisit previously proposed policies and rigorously establish their throughput and delay-optimality. We propose a *Policy Splicing* technique to combine scheduling policies for small networks in order to construct throughput-optimal policies for larger networks, some of which also aim for low delay. For $N = 3$, there exists a sum-queue length optimal scheduling policy that requires only the empty-nonempty state of each queue. We show, however, that for $N \geq 4$, there is no scheduling policy that uses only the empty-nonempty states of the queues and is sum-queue length optimal over all arrival rate vectors in the capacity region.

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. In the SoC setting, we propose a throughput-optimal policy that only uses information that nodes in the network can glean by sensing activity (or lack thereof) on the channel.

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. Our study throws up some new insights for the above classes of networks:

- knowledge of queue length information is not necessary to achieve optimal throughput/delay performance for certain classes of interference constraints,
- for networks in these classes, it is possible to perform throughput-optimal scheduling by merely knowing whether queues in the network are empty or not, and
- it is also possible to be throughput-optimal by *not* always scheduling the maximum possible number of nonempty non-interfering queues.

Index Terms

Wireless Sensor Networks, Medium Access Control (MAC) protocols, Optimal Polling, Delay Minimization, Hybrid MACs, Self-Organizing Networks, Internet of Things (IoT).

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 large-scale wireless sensor network comprising low-cost sensors that forward measurements from their respective locations. Given the massive scale and ubiquitous nature of these wireless networks, for IoT solutions to be viable, the embedded IoT devices, or *motes* (as they are often called) will naturally have to cost very little (less than \$1, according to some estimates [1]). Such devices will

- *have to consume very low power* since a sensor in an IoT network is typically expected to last for several years without replacement. Battery and replacement expenses can affect the cost of deployment and maintenance adversely. [2, Table III] gives quite an extensive list of battery lifetimes expected from motes in various IoT applications.
- *possess very limited communication capabilities*. This is, in fact, the first of two important consequences of the low power consumption constraint. Frequent communication with a centralized coordinating entity (such as a Path Computation Entity, or PCE, in the 6TiSCH architecture [3]) will place unnecessary burdens on an already energy limited device.

Avinash Mohan is with the Technion Israel Institute of Technology, Haifa-3200003, Israel. Aditya Gopalan and Anurag Kumar are with the Indian Institute of Science, Bangalore-560012, KA, India. e-mail: avinashmohan@campus.technion.ac.il, {aditya, anurag}@iisc.ac.in, respectively. *Avinash Mohan is the corresponding author.*

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 last author.

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

Decentralized MAC protocols, wherein the nodes autonomously take transmission decisions based on limited information about the network state¹ will, therefore, need to be used instead of those that employ centralized control of transmissions.

- *possess very limited memory and computing power.* This is the second consequence of the power consumption constraint. To keep cost-per-device and energy drain low, one cannot equip such a sensor with anything beyond the bare minimum memory and processing abilities required for transmitting small packets based on a simple-to-compute transmission schedule.

The above requirements paint a picture of a *highly resource-challenged network*. Furthermore, these constraints are starkly different from those that are encountered in the transmission of traditional voice and packet data over wireline or cellular wireless networks. In cellular systems, for example, the preponderance of scheduling decisions comes from the base station and hence, control is centralized [4, Chapters 6, 13], but some Quality of Service (QoS) is expected – low packet delay, for instance [5], [6]. In contention access systems such as WiFi, scheduling is distributed but the service is best effort [7] or, at best, differentiated – such as with the “Enhanced Distributed Channel Access” [8] and the “Enhanced Distributed Coordination Function” (E-DCF) mechanisms in the IEEE 802.11e standard [9]. In many IoT applications, e.g., condition monitoring or predictive maintenance, there is a need for distributed scheduling (for the earlier reasons) while also providing QoS. Consequently, resource allocation techniques developed to handle packet and voice data are insufficient to address the aforementioned issues in resource challenged networks. Most existing medium access protocols and scheduling algorithms suffer from limitations such as requiring too much state information to compute the schedule in any time slot making them hard to decentralize, being computationally intensive and thus unsuited for implementation on low-cost IoT devices, etc. Our aim in the paper is to propose decentralized 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 Tassioulas and Ephremides [10]. This paper introduces the model of 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. The pair-wise constraints are represented by a *conflict graph* (also known as an *interference graph*) with the queues as the nodes and the pair-wise scheduling constraints being the edges. 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, and show that it is throughput-optimal, i.e., it stochastically stabilizes all queues under any stabilizable arrival rate.

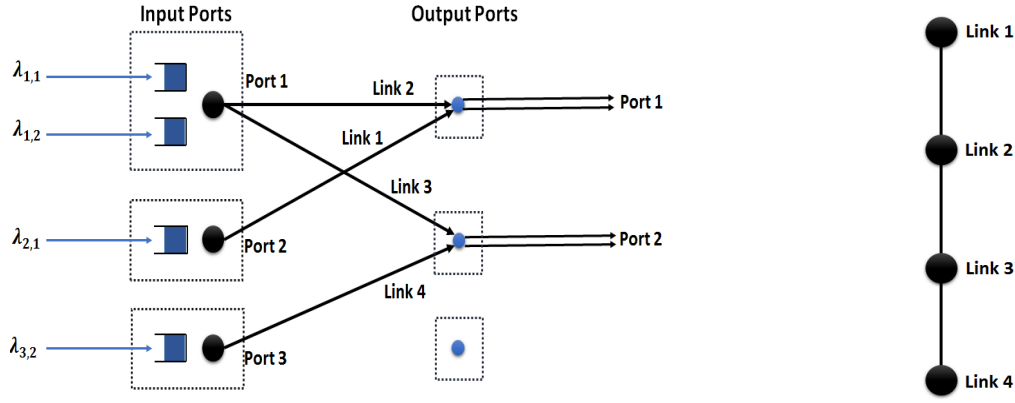
Attempts to decentralize MaxWeight include approximations based on message passing between nodes [11], [12], or using queue lengths to modulate backoff parameters in CSMA and ALOHA [13], [14]. Both of these methods, while being throughput-optimal, suffer from poor delay performance. Another method to reduce the amount of information required for scheduling is proposed in [15], where, for two classes of constrained queueing systems, algorithms relying only on the empty-nonempty state of queues is proposed and analysed for delay performance. Our interest lies in the second half of [15], where a scheduling algorithm is proposed for a system of N parallel queues in which adjacent queues cannot be served simultaneously. The authors give the delay optimal policy for $N = 3$. This has been extended to $N = 4$ by Ji et al [16] where the heavy-traffic delay-optimality of the proposed policy was proved. One of the contributions of our work in this paper is to refine these earlier contributions, and then to provide a novel method (policy *splicing*) to develop scheduling algorithms for larger numbers of nodes. It is not yet clear if it is even possible to extend these algorithms to general wireless networks while preserving performance guarantees such as throughput-optimality.

A third class of strategies has focused on completely uncoordinated medium access, in contrast to methods using network state information. Here the focus is on improving the saturation throughput of Abramson’s ALOHA protocol beyond $(1/e)$, without any queue length knowledge. The main idea is to allow collisions to take place, but use physical layer techniques like successive interference cancellation to decode the garbled messages over multiple time slots [17], [18]. These techniques, like Abramson’s ALOHA, allow lightweight uncoordinated access, but are not throughput-optimal, in that they can hope to achieve only the saturation throughput (the maximum sum rate point in the stability region). In contrast, we are interested in contention-free, low coordination and throughput-optimal MAC schemes.

One other important contribution of this paper comes from a purely theoretical standpoint. As opposed to colocated networks, for which it is well-known that the empty-nonempty statuses of queues is sufficient for stability and good delay performance, it is a common perception that queue length information is required to achieve throughput-optimality in non-colocated networks. The authors in [19] provide an example which helps reinforce this point: let us consider the classical input-queued switch (shown below in Fig. 1). As can be seen from the associated conflict graph in Fig. 1, the arrival rate to the queue associated with Link 1 is $\lambda_{2,1}$, to Link 2 is $\lambda_{1,1}$, to Link 3 is $\lambda_{1,2}$ and to Link 4 is $\lambda_{3,2}$. The backlog of the queue associated with Link i is denoted $Q_i(t)$, where we continue with the queue-length, arrival and departure embedding that we have been using until now. Activation constraints prevent scheduling of adjacent links, which means that adjacent queues cannot be served simultaneously and hence,

$$\lambda_{2,1} + \lambda_{1,1} < 1, \quad (1)$$

¹Like, perhaps, only the information they can gather by sensing their immediate surroundings, i.e, one-hop neighborhood.



(a) A 3×3 input-queued switch with only 4 active flows. $\lambda_{i,j}$ denotes the rate of arrival of packets to input port i and headed for output port j .

(b) The associated conflict (interference) graph is a path graph.

Fig. 1: An input-queued switch and the associated interference graph.

$$\lambda_{1,1} + \lambda_{1,2} < 1, \text{ and} \quad (2)$$

$$\lambda_{1,2} + \lambda_{3,2} < 1. \quad (3)$$

As discussed in Sec. II, this is, in fact, the capacity region of the network in Fig. 1, which means that for every rate vector $[\lambda_{1,1}, \lambda_{1,2}, \lambda_{2,1}, \lambda_{3,2}]$ in the above region (i.e., satisfying (3)), there exists a scheduling policy that ensures stability. Now, consider a randomized scheduling policy that schedules the largest number of non interfering links in every slot and ties are broken with equal probability. In particular, this means that

- In every slot, the policy chooses between the following three pairs: Queues 1 and 3, Queues 2 and 4 and Queues 1 and 4. Also,
- If, for example, only Queues 1, 3 and 4 are nonempty, the policy serves Queues 1 and 3 w.p. $\frac{1}{2}$, and Queues 1 and 4 w.p. $\frac{1}{2}$.

Assume that the arrival process to each queue is IID across time and Bernoulli. Let $\lambda_{i,j} = 0.5 - \delta$, where $\delta > 0$, i.e., arrival rates are all equal, and note that this satisfies the constraints in (3). Now consider input port 1 which, in the interference graph, is the combination of Queues 2 and 3. The total arrival rate to this pair = $\lambda_{1,1} + \lambda_{1,2} = 1 - 2\delta$. In any slot with $Q_2(t) > 0$, and $Q_3(t) > 0$, since arrivals to Queues 1 and 4 are Bernoulli, they will both be non empty w.p. $(0.5 - \delta)^2$, which means that the $Q_2 + Q_3$ pair will receive service w.p. $\frac{2}{3}$. Therefore, the per-slot probability that this pair receives service is upper bounded by

$$\frac{2}{3} \times (0.5 - \delta)^2 + 1 \times \underbrace{\left(1 - (0.5 - \delta)^2\right)}_{\text{At least one of Queue 1 or 4 received 0 arrivals}} \quad (4)$$

$$< 1 - 2\delta = \lambda_{1,1} + \lambda_{1,2}, \quad \forall \delta < 0.0358, \quad (5)$$

which proves that this policy clearly renders input port 1, unstable. Of particular note is the fact that this policy does not attempt of break ties in favor of the two inner queues that are more constrained (Queue 2 cannot be served if either Queue 1 or Queue 3 is being served) than the two outer queues. As we will show in the sequel, prioritizing inner queues will have a strong impact on both stability and delay. Any policy that schedules the largest number of non-empty non-interfering queues in every slot is called a *Maximum Size Matching* (MSM) policy and the one analysed above is an example of this class. One obvious candidate for a throughput-optimal scheduling policy is MaxWeight. We will, however, show that the MaxWeight policy is *not desirable* in this case due to two reasons

- The first being that it violates our requirement of low state information exchange for taking scheduling decisions.
- The second being that MaxWeight is provably *non delay optimal* over these types of interference graphs.

Another obvious candidate is the aforementioned class of QCSMA, i.e., queue length-modulated CSMA, policies ([13], [14]), which, although stabilizing and amenable to distributed implementation shows unacceptable delay performance, as discussed before. These two observations combined with the instability of the MSM policy bring forth an important question: is it truly possible to stabilize non-located networks and achieve low queueing delay without using queue lengths? In the sequel, we will answer this question in the affirmative.

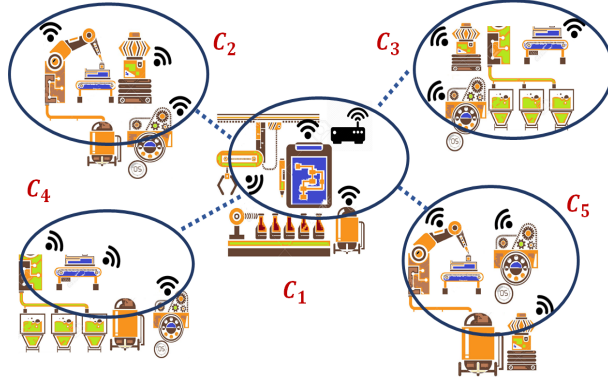


Fig. 2: A wireless sensor network on a factory floor showing multiple workstations. Sensors within a workstation form a clique or a collocated network. The network in the figure comprises $N = 5$ cliques. Fig. 5b shows the associated conflict graph.

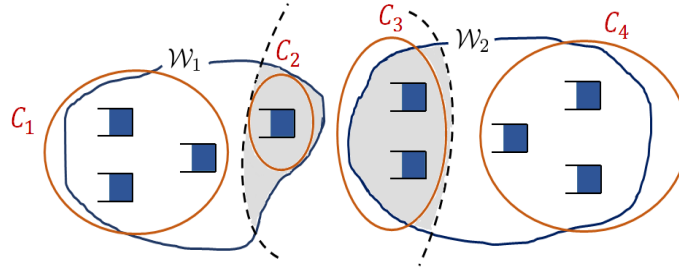


Fig. 3: An example of an LAoC. The communication links in the two regions **shaded gray** interfere with each other. Since each of the two networks \mathcal{W}_1 and \mathcal{W}_2 is collocated, links within them interfere as well. The total network can hence, be decomposed into 4 cliques, viz. C_1, \dots, C_4 . Fig. 5a shows the associated conflict graph.

A. Our Contributions and Organization

It is well known that collocated networks admit stable and delay-optimal scheduling policies that require only the empty-non empty states of constituent queues. In this work, we first develop *centralized* throughput-optimal and low-delay² scheduling policies that rely only on reduced state information, namely the empty-nonempty states of queues. Thereafter, we use these policies to construct reduced-state *decentralized* scheduling protocols for multiple classes of networks. Our specific contributions are as follows.

- We begin by studying scheduling of transmissions on “path-graph interference networks.” We restrict ourselves to the subclass of Maximum Size Matching policies (MSM) mentioned above, that additionally require only the empty-nonempty statuses of the queues therein (introduced in Sections III and IV) and will provide a complete characterization of the set of such policies for the case with $N = 3$ queues (Sec. V). 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, that MaxWeight is *not* delay-optimal in such networks in a stochastic ordering sense.
- Continuing with path-graph networks, we propose a “policy splicing” technique (see Figures 9 and 10) to combine policies for small networks to give rise to policies for larger networks (Sections VII and VIII.). We use this technique to propose MSM scheduling policies for several such networks. We also provide an in-depth analysis of delay with MSM policies (Sec. IX) culminating in a result that shows that there do not exist delay optimal MSM policies for such networks with $N \geq 4$ queues (Thm. 21).
- We then extend our theory of MSM policies to schedule transmissions over a much more general class of networks that we call “Cluster-of-Cliques Constraint Networks,” such as the ones in Figures 2 and 3 (Sec. X). We will also see multiple methods to further reduce the amount of state information (empty-nonempty statuses of the queues) 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 [20], 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. XI).
- We then present numerical results (Sec. XII) showing the performance of our proposed policies, and comparisons with standard, high-overhead state-based policies such as the MaxWeight- α family [21].

²By low delay, we mean a low sum queue length across the system.

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 [10]. 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 $0, 1, 2, \dots$. Exogenous arrivals to the queues are embedded *at* slot boundaries, $t = 0, 1, 2, \dots$, 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 iid³ 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. X. We use $\mathbf{Q}(t) = [Q_1(t), \dots, Q_N(t)]$ 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) time⁴, and any scheduling overhead. Packet transmissions are assumed to take exactly one time slot and succeed with probability⁵ 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, (7) gives the constraints for path-graph interference networks and (9) for Star-of-Cliques networks. The vector $\mathbf{S}(t) := [S_1(t), \dots, S_N(t)]$ satisfying the interference constraints is called an *activation vector*. Thus, for every queue i ,

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

Denote by $\zeta(t) := [\mathbb{I}_{\{Q_1(t) > 0\}}, \dots, \mathbb{I}_{\{Q_N(t) > 0\}}]$ 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 $\pi := \{\mu_0, \mu_1, \dots\}$ 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, $\mu_t : \mathcal{H}_t \rightarrow \mathcal{V}$ 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 $\pi = \{\mu, \mu, \dots\}$, 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

$$\limsup_{T \rightarrow \infty} \frac{1}{T} \sum_{t=0}^{T-1} \sum_{i=1}^N \mathbb{E}_{\mathbf{Q}(0)}^{(\pi)} Q_i(t) < \infty. \quad (6)$$

This condition is commonly known as *strong stability* [22]. A policy that ensures (6) 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 [10], 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. 4). The scheduling constraints are the same as the second model in Tassiulas and Ephremides 1994 [15], 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 $\mathbf{S}(t)$, $\forall t \geq 0$

$$S_i(t) + S_{i+1}(t) \leq 1, \quad \forall t \geq 0, 1 \leq i \leq N - 1. \quad (7)$$

The conflict graph associated with the system is called *path graph* [23], [24]. Standard analysis [10] show that the capacity

³“iid” stands for *independent and identically distributed*.

⁴Most wireless systems require a MAC layer acknowledgement to combat high packet error rates

⁵The effects of fading and channel errors are not considered here and are a subject of future research.

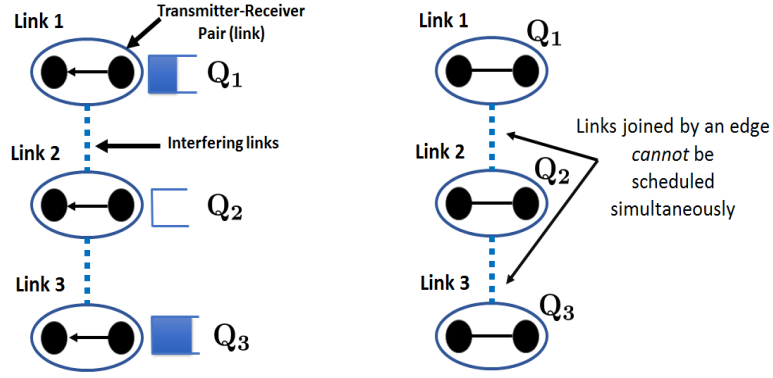


Fig. 4: The basic path-graph interference system with $N = 3$ communication links along with the associated packet queues (left) and its conflict graph (right). The interference constraints are such that physically adjacent queues cannot be served simultaneously.

region of this network is the set

$$\Lambda_N := \{\lambda \in \mathbb{R}_+^N \mid \lambda_i + \lambda_{i+1} \leq 1, \forall 1 \leq i \leq N-1\}, \quad (8)$$

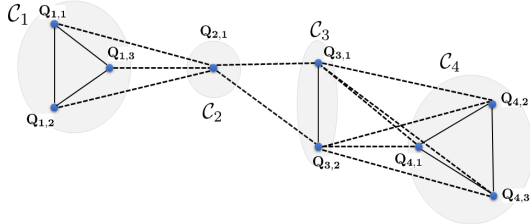
whose interior, Λ_N^o , is the set of all stabilizable rate vectors.

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

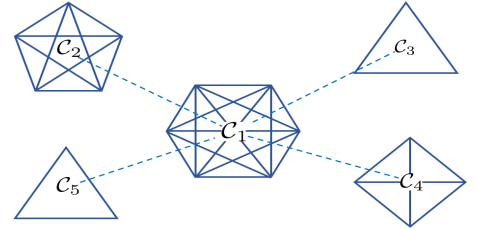
In the remainder of the paper, we will refer to the conflict graph associated with a collocated network, i.e., a fully connected graph or subgraph, as a *clique*. The system under consideration comprises multiple cliques and the exact nature of the interference relations *across* cliques are described 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



(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. 5: Cluster-of-Cliques networks.

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. 5b). In other words, the network's conflict graph consists of N cliques denoted C_1, \dots, C_N , and clique C_i consists of 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

$$S_{i,j}(t) + S_{k,l}(t) \leq 1, \forall t \geq 0, \text{ if } i = k, \text{ or } i = 1, \text{ or } k = 1. \quad (9)$$

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

$$\begin{aligned} \Lambda_s^{(N)} &:= \left\{ \lambda \in \mathbb{R}_+^N \mid \sum_{j=1}^{N_1} \lambda_{1,j} \right. \\ &\quad \left. + \sum_{k=1}^{N_i} \lambda_{i,k} \leq 1, i \in \{2, \dots, N\} \right\} \end{aligned} \quad (10)$$

(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, \dots, C_N , but unlike the SoC model, all transmissions in C_{i-1} interfere with those in C_i , $i \in \{2, \dots, N\}$ and vice-versa (see Fig. 5a). As in the SoC model, Clique C_i comprises N_i queues and $N \equiv \sum_{i=1}^N N_i$ 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

$$S_{i,j}(t) + S_{k,l}(t) \leq 1, \quad \forall t \geq 0, \text{ if } k = i + 1, \quad \forall 1 \leq i \leq N - 1. \quad (11)$$

The capacity region of this system is given by

$$\Lambda_l^{(N)} := \left\{ \lambda \in \mathbb{R}_+^N \left| \sum_{j=1}^{N_i} \lambda_{i,j} + \sum_{k=1}^{N_{i+1}} \lambda_{i+1,k} \leq 1, \quad i \in \{1, \dots, N-1\} \right. \right\} \quad (12)$$

(the subscript l highlights the fact that this is the Linear-Array-of-Cliques model). As before, the vector $\mathbf{S}(t) := [S_1(t), \dots, S_N(t)] \in \{0, 1\}^N$ is called an *activation vector* if it satisfies the constraints in (9) and (11) in the SoC and LAoC systems, respectively. We now begin our study with path graph interference networks.

III. MAXIMUM SIZE MATCHING (MSM) POLICIES

We first define the subclass of scheduling policies to which we will restrict our attention and provide some motivation to do so in from the perspective of delay reduction. The latter will be stated and explained more formally in subsequent sections.

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

For example, if $N = 7$, and $\mathbf{Q}(t) = [1, 2, 0, 0, 4, 3, 3]$, 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, [15] shows 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.

Notice that in Fig. 4, Queue 2 cannot be served in any slot in which either Queue 1 or Queue 3 is being served and similarly Queue 3 cannot be served in any slot in which either Queue 2 or Queue 4 is being served. In contrast, service to Queues 1 depends only on whether Queue 2 is being served, which makes it less constrained from the perspective of service. 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,” more constrained queues. Lemma 4.1 in [15], which we state below (Lem. 1) since we will be invoking it often in the sequel, 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 (Conditions 1 & 2 below), and
- 2) the policy should prioritize *inner* queues over *outer* queues while breaking ties (Condition 3 below).

Specifically, in [15], the authors provide a sufficient but *not necessary* condition for an activation vector to serve the largest number of nonempty queues in a slot. Define $S(\zeta(t)) \subset V$ as the set of all activation vectors that serve the largest number of queues in slot t when the occupancy vector is $\zeta(t)$. Given an occupancy vector ζ , let $k = k(\zeta)$ be the *twice* the number of runs of nonempty queues, and $j_1 = j_1(\zeta), \dots, j_k = j_k(\zeta)$, the nonempty queues that mark the beginnings ($j_{\{\text{odd subscript}\}}$) and ends ($j_{\{\text{even subscript}\}}$) of the nonempty runs, or the two extreme queues (Queues 1 and N). Fig. 6 illustrates this numbering scheme. Clearly,

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

Lemma 1. (Lem. 4.1 in [15]) $\mathbf{S}(t) \in S(\zeta(t))$ if

- 1) **Odd-length run condition:** If $j_{2m} - j_{2m-1}$ is even, then for all $j_{2m-1} \leq j \leq j_{2m}$,

$$S_j(t) = \begin{cases} 1 & \text{if } j - j_{2m-1} \text{ is even,} \\ 0 & \text{otherwise,} \end{cases}$$

$$m = 1, \dots, k/2.$$

- 2) **Even-length run condition:** If $j_{2m} - j_{2m-1}$ is odd, then any one of the following 3 conditions must be satisfied

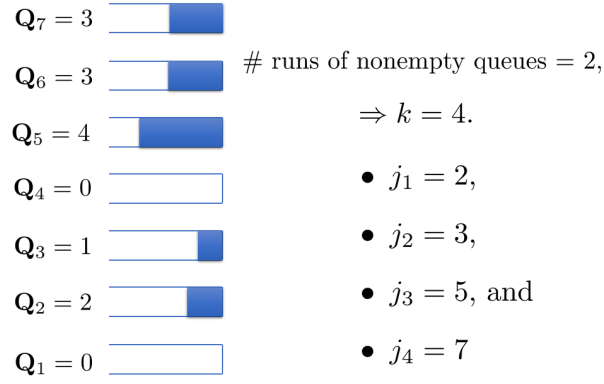


Fig. 6: Figure depicting how runs of non empty queues are numbered. Here, $N = 7$ and since $\zeta = [0, 1, 1, 0, 1, 1, 1]^T$, $S(\zeta) = \{[0, 1, 0, 0, 1, 0, 1]^T, [0, 0, 1, 0, 1, 0, 1]^T\}$. There are two runs of nonempty queues, the first beginning at Queue 2 and ending at Queue 3, and the second 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.

a) For every $j_{2m-1} \leq j \leq j_{2m}$,

$$S_j(t) = \begin{cases} 1 & \text{if } j - j_{2m-1} \text{ is even,} \\ 0 & \text{otherwise,} \end{cases}$$

b) For every $j_{2m-1} \leq j \leq j_{2m}$,

$$S_j(t) = \begin{cases} 1 & \text{if } j - j_{2m-1} \text{ is odd,} \\ 0 & \text{otherwise,} \end{cases}$$

or,

c) There exists an l such that

$$S_j(t) = \begin{cases} 1 & \text{if } j - j_{2m-1} \text{ is even and } j_{2m-1} \leq j < l, \\ & \text{or } j_{2m} - j \text{ is even and } j_{2m} \geq j > l + 1, \\ 0 & \text{otherwise,} \end{cases}$$

$$m = 1, \dots, k/2.$$

3) **Inner queues priority condition:** If $j_1 = 1$,

$$S_j(t) = \begin{cases} 1 & \text{if } j_2 - j \text{ is even, } j_1 \leq j \leq j_2, \\ 0 & \text{otherwise,} \end{cases}$$

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 $S(\zeta)$. We will see, later that the third condition helps reduce delay by prioritizing “inner” queues. With this, one simply needs to ensure that in every slot, the policy chooses activation vectors only from $S(\zeta(t))$, to ensure that it is MSM. Note that there might exist several MSM activation vectors for a policy to choose from, in a given slot. The ultimate choice might depend on all of \mathcal{H}_t (the history) and not only on $\zeta(t)$. MaxWeight (*MW*) is the obvious example here, since it uses $\mathbf{Q}(t)$ for scheduling rather than just $\zeta(t)$. So, for the same $\zeta(t)$, MaxWeight could end up choosing different MSM vectors, depending on the actual queue lengths in those slots. But we will show that in several interference graphs, $\zeta(t)$ is sufficient not only for stability but also for delay optimality.

Notation: Classes of scheduling policies (see Fig. 7)

- $\Pi^{(N)}$: the class of all policies.
- $\Gamma_M^{(N)}$: the class of all MSM policies.
- $\Pi_M^{(N)}$: the class of all policies that take only the occupancy vector $\zeta(t)$ 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.
- $\tilde{\Pi}^{(N)}$: the class of all MSM policies within $\Pi_M^{(N)}$ that additionally break ties in favour of inner queues (see condition 2 above).

Note that $\Pi^{(N)} \supseteq \Gamma_M^{(N)} \supseteq \Pi_M^{(N)} \supseteq \tilde{\Pi}^{(N)}$. Going back to our 7-queue example, when $\zeta(t) = [1, 1, 0, 0, 1, 1, 1]$, 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 $\tilde{\Pi}^{(7)}$.

IV. QUEUE NONEMPTINESS-BASED (QNB) SCHEDULING

While almost all well-known policies use full queue length-information ($\mathbf{Q}(t)$) to take scheduling decisions, e.g., MaxWeight [10], a key objective of this paper is to show that for many classes of interference graphs, throughput-optimal policies can be designed that use much less information. By “queue nonemptiness-based policies,” we mean those that only require the knowledge of the occupancy vector, i.e., $\zeta(t)$. Clearly, this contains much less information than the vector $\mathbf{Q}(t)$ that MaxWeight requires, and $\zeta(t)$ can be transmitted across the network with just 1 bit per queue per slot. The functions $\{\mu_t, t \geq 0\}$ that constitute the policy are now maps of the form $\mu_t : \{0, 1\}^N \rightarrow \mathcal{V} \subseteq \{0, 1\}^N$, 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, the delay properties of such a reduced state scheduler are naturally suspect, 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 [25].

We now provide a sufficient condition that will later help construct strongly stable policies that use only $\{\zeta(t), t \geq 0\}$, by proving a Lyapunov drift result that will be invoked often in the sequel.

Lemma 2. Consider the class of systems described in Section II, and define property \mathcal{P} as

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

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 $\zeta(t)$. In words, (\mathcal{P}) reads: “for a pair of neighboring queues, there is no departure from either of these queues *iff* 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 $\zeta(t) = (1, 1, 1, 1)$, $\mathbf{S}(t) = (1, 0, 1, 0)$ satisfies condition (\mathcal{P}) , but $\mathbf{S}(t) = (1, 0, 0, 1)$ does not.

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

$$L(\mathbf{Q}(t)) := \sum_{i=1}^{N-1} (Q_i(t) + Q_{i+1}(t))^2 \quad (13)$$

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. XIV-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 $\mathbf{Q} = [Q_1, \dots, Q_N]$.

$$\begin{aligned} & \mathbb{E}[L(t+1) - L(t) \mid \mathbf{Q}(t) = \mathbf{Q}] \\ &= \sum_{i=1}^{N-1} \mathbb{E} \left[(Q_i - D_i + A_i + Q_{i+1} - D_{i+1} + A_{i+1})^2 \right. \\ & \quad \left. - (Q_i + Q_{i+1})^2 \mid \mathbf{Q}(t) = \mathbf{q} \right] \\ & \stackrel{*}{\leq} \sum_{i=1}^{N-1} \left[(Q_i + Q_{i+1})^2 + 1 + 4 - 2(Q_i + Q_{i+1}) \right. \\ & \quad \left. (\mathbb{E}[D_i + D_{i+1} \mid \mathbf{Q}] - \lambda_i - \lambda_{i+1}) - (Q_i + Q_{i+1})^2 \right] \\ &= \sum_{i=1}^{N-1} \left[5 - 2(Q_i + Q_{i+1}) \right. \\ & \quad \left. (\mathbb{E}[D_i + D_{i+1} \mid \mathbf{Q}(t) = \mathbf{Q}] - \lambda_i - \lambda_{i+1}) \right], \end{aligned} \quad (14)$$

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. (14), and thus *removing conditioning*, we get

$$\begin{aligned} \mathbb{E}[L(t+1) - L(t)] &\leq \sum_{i=1}^{N-1} \left[5 - 2\mathbb{E}((Q_i + Q_{i+1}) \right. \\ &\quad \times \mathbb{E}[D_i + D_{i+1} \mid \mathbf{Q}(t)]) \\ &\quad \left. + 2(\lambda_i + \lambda_{i+1})\mathbb{E}(Q_i + Q_{i+1}) \right] \end{aligned} \quad (15)$$

We now use the fact that the policy satisfies property \mathcal{P} , to see that $\mathbb{E}[D_1 + D_2 \mid \mathbf{Q}(t)] = \mathbb{I}_{\{Q_1 + Q_2 > 0\}}$, w.p.1, and the fact that for any non negative random variable Z , $\mathbb{E}(Z\mathbb{I}_{\{Z>0\}}) = \mathbb{E}Z$, whereby,

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

This means that

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

where $\epsilon_i = 1 - \lambda_i - \lambda_{i+1}$. Note that from the definition of Λ° , $\epsilon_i > 0$, $\forall 1 \leq i \leq N-1$. Substituting this in Eqn. (15), we get

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

where in inequality \dagger , $\epsilon := \min_{1 \leq i \leq N-1} \epsilon_i$. 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

$$\mathbb{E}[L(t+1) - L(t)] \leq 5(N-1) - 2\epsilon \sum_{i=1}^N \mathbb{E}Q_i(t). \quad (16)$$

Using the telescoping sum technique (see [22]) it can now be shown that the process $\{\mathbf{Q}(t), t \geq 0\}$, is strongly stable. ■

V. 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 $\Pi^{(3)}$. Since this part of the thesis uses heavy notation, for the reader's convenience a glossary of notation is provided here: XIV-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 $\zeta(t)$, of which there are 8 alternatives, to one of these two activation vectors, giving us $2^8 = 256$ nonemptiness-based policies in all. Suppose $|A|$ denotes the cardinality of set A . We prove that upon imposing the MSM condition, this number reduces to 4, i.e., $|\Pi_M^{(3)}| = 4$ as follows. There is no choice to be made when $\zeta(t)$ is either $\mathbf{0}$, a singleton, or $[1, 0, 1]$. The MSM condition also means that $\mathbf{S}(t) = [1, 1, 1] \mapsto \zeta(t) = [1, 0, 1]$. This leaves only two states, viz., $[1, 1, 0]$ and $[0, 1, 1]$, each of which can be assigned either $[0, 1, 0]$ or $[1, 0, 1]$ and hence, $|\Pi_M^{(3)}| = 4$.

Characterization of $\Pi_M^{(3)}$: We now show that this class contains throughput optimal, delay optimal, and also *unstable* MSM policies. First, some additional notation is in order. Depending on the mapping from $\zeta(t)$ 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 thesis.

- The subscripts “TD” and “BU” stand for “Top-Down” and “Bottom-Up,” respectively and the reason for this nomenclature will become apparent shortly.
- A “~” 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, $\tilde{\pi}_{IQ}^{(3)} \in \tilde{\Pi}^3$.

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

$\zeta = [\zeta_1(t), \zeta_2(t), \zeta_3(t)]$	$\pi_{TD}^{(3)}$	$\pi_{BU}^{(3)}$	$\tilde{\pi}_{IQ}^{(3)}$	$\pi_{OQ}^{(3)}$
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, and the caption for Fig. 7 specifies to which class each of these four policies belongs. In what follows we will describe and analyse each of these policies in detail. Notice from the entries corresponding to the rows $\zeta = [011]$ and $\zeta = [110]$ that $\pi_{TD}^{(3)}$ and $\pi_{BU}^{(3)}$ are complementary policies, and so are $\tilde{\pi}_{IQ}^{(3)}$ and $\pi_{OQ}^{(3)}$. Specifically, each of these four policies induces the following priority order, which will become clear when we consider each of them individually later:

- $\pi_{TD}^{(3)}$ gives *decreasing* priority to Queues 1, 2 and 3 in that order,
- $\pi_{BU}^{(3)}$ gives *increasing* priority to Queues 1, 2 and 3 in that order,
- $\tilde{\pi}_{IQ}^{(3)}$ gives maximum priority to Queue 2 the *inner* queue (once again, check the rows in Table. I corresponding to $\zeta = [011]$ and $\zeta = [110]$), 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
- $\pi_{OQ}^{(3)}$ prioritizes the two *outer* queues.

To begin with, we show that $\pi_{TD}^{(3)}$ and $\pi_{BU}^{(3)}$ 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.

A. Analysis of $\pi_{TD}^{(3)}$ and $\pi_{BU}^{(3)}$

As the column corresponding to $\pi_{TD}^{(3)}$ in Table. I shows, this policy clearly gives absolute priority to Queue 1, i.e., serves Queue 1 whenever it is nonempty, and can be restated as follows.

At time t

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

In words, the policy $\pi_{TD}^{(3)}$ simply reads “prioritize Queue 1 over Queue 2, and Queue 2 over Queue 3, while scheduling all possible non-interfering queues.” Hence, the subscript “TD,” since this policy, in a sense, establishes a “Top-Down” priority.

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

The proof of Theorem 3 uses the fact that under $\pi_{TD}^{(3)}$, 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. $\pi_{BU}^{(3)}$ simply swaps the priorities of Queues 1 and 3 and its proof proceeds mutatis mutandis. The complete proof is available in Sec. XIV-D in the Appendix.

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

This policy can be restated as follows.

At time t :

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

In [15], it has been asserted without formal proof that $\tilde{\pi}_{IQ}^{(3)}$ is delay optimal. We begin analysing the policy by proving that it is Throughput Optimal.

Theorem 4. $\tilde{\pi}_{IQ}^{(3)}$ is throughput-optimal.

The proof of this result involves showing that $\tilde{\pi}_{IQ}^{(3)}$ satisfies property \mathcal{P} in Lem. 2 and is therefore T.O. The proof is available in Sec. XIV-E in the Appendix.

We next turn to the delay performance of the policy $\tilde{\pi}(3)_{IQ}$. Tassiulas and Ephremides [15, Theorem 4.2] define a projection operator $L : \Pi^{(N)} \rightarrow \Gamma_M^{(N)}$ that takes any policy $\pi \in \Pi^{(N)}$ and produces an MSM policy, $L(\pi)$. They then show 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 Theorem 4.2 in [15] 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

$$\sum_{i=1}^N Q_i^{L(\pi)}(t) \leq^{st} \sum_{i=1}^N Q_i^\pi(t), \quad \forall t \geq 0, \quad (17)$$

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*:

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 $\lambda \in \Lambda_N$,

$$\sum_{i=1}^N Q_i^{\pi^*}(t) \leq^{st} \sum_{i=1}^N Q_i^\pi(t), \quad \forall t \geq 0. \quad (18)$$

In [15, Remark 2, pp. 353], it is suggested that for $N = 3$ there is exactly one MSM policy and that, as a result of Theorem 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 [15] is $\tilde{\pi}_{IQ}^{(3)}$, which also prioritises inner queues. However, the projection operator $L(\cdot)$ does not ensure the inner queue prioritisation condition (3 in Lemma 4.1 therein). Thus, Theorem 4.2, as it stands, merely asserts that any one of the MSM policies could be delay optimal. It requires a further step in the proof to show that $\tilde{\pi}_{IQ}^{(3)}$ is, indeed, the unique uniformly delay optimal policy for $N = 3$. We proceed to do so below.

C. Improving Delay Performance via Projections

We use the operator $L : \Pi^{(N)} \rightarrow \Gamma_M^{(N)}$ (defined in [15]) that takes any policy $\pi \in \Pi^{(N)}$ and produces an MSM policy, $\pi_M \equiv L(\pi)$. In slot t , suppose the occupancy vector is $\zeta(t)$ and the system backlog is $\mathbf{Q}(t)$. Suppose also that π chooses the activation vector \mathbf{s}' in the slot. The operator L produces another policy π_M by constructing its activation vector \mathbf{s} , as follows.

Definition. [15, Lem. 4.2] Given $\pi \in \Pi^{(N)}$ and occupancy vector ζ ,

- 1) For all odd-length runs $\{j_{2m-1}, \dots, j_{2m}\}$ of nonempty queues π_M activates Queue j_{2m-1} and every *other* queue until and excluding j_{2m} .
- 2) When the run is of even length, π_M follows π in the following sense.
 - a) If π does not choose Queue j_{2m-1} , so does π_M , i.e., if $s'_{j_{2m-1}} = 0$

$$s_j(t) = \begin{cases} 1 & \text{if } j - j_{2m-1} \text{ is odd,} \\ 0 & \text{otherwise,} \end{cases}$$

- b) If π chooses Queue j_{2m-1} , so does π_M , i.e., if $s'_{j_{2m-1}} = 1$

$$s_j(t) = \begin{cases} 1 & \text{if } j - j_{2m-1} \text{ is even,} \\ 0 & \text{otherwise,} \end{cases}$$

- c) Finally, if $s'_{j_{2m-1}} = 1$ and $s'_{j_{2m}} = 1$, there must exist⁶two consecutive 0 entries in \mathbf{s}' between j_{2m-1} and j_{2m} . Let l be the smallest such number, i.e., $s'_l = s'_{l+1} = 0$. Then

$$s_j(t) = \begin{cases} 1 & \text{if } j - j_{2m-1} \text{ is even, and } j_{2m-1} \leq j < l \\ & \text{or } j_{2m} - j \text{ is even, and } l + 1 < j \leq j_{2m} \\ 0 & \text{otherwise,} \end{cases}$$

Suppose we denote by $\mathbf{Q}^{\pi_M}(t)$ the backlog induced by π_M . Thm. 4.2 in [15] shows that when the systems upon which π and π_M act are started out in the same initial state and the arrivals have the same statistics,

$$\sum_{i=1}^N Q_i^{\pi_M}(t) \leq^{st} \sum_{i=1}^N Q_i^\pi(t). \quad (19)$$

⁶If not, one would begin with $s'_{j_{2m-1}} = 1$ and set every *other* $s'_j = 1$ resulting in $s'_{j_{2m}} = 0$, which results in a contradiction.

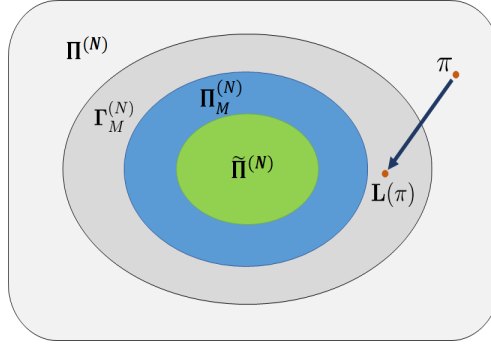


Fig. 7: Illustrating the four policy spaces and the action of the projection operator L . Recall that $\pi_{TD}^{(3)}, \pi_{BU}^{(3)}$, and $\pi_{OQ}^{(3)} \in \Pi_M^{(3)}$, and that $\tilde{\pi}_{IQ}^{(3)} \in \tilde{\Pi}^{(3)}$. In Sec. V-E we will encounter a policy, $\pi_{IQ}^{(3)} \in \Pi^{(3)} \setminus \Gamma_M^{(3)}$.

It is easy to see that L is a projection onto $\Gamma_M^{(N)}$, i.e.,

- $L^2 := L \circ L = L$, which means that $L(L(\pi)) = L(\pi)$, $\forall \pi \in \Pi^{(N)}$, and
- On $\Gamma_M^{(N)}$, L is the identity map, i.e., for every $\pi_M \in \Gamma_M^{(N)}$, $L(\pi_M) = \pi_M$.

Remark. What is most important about this projection is that it does *not* map directly to $\tilde{\Pi}^{(N)}$. Specifically, L does not guarantee that the *second* MSM condition (condition 2) in Sec. III is satisfied by $L(\pi)$ for any $\pi \in \Pi^{(N)}$. In other words, L does not necessarily lead to an MSM policy that prioritises inner queues. Condition 3 in Lem. 1 is necessarily satisfied by $L(\pi)$ for any $\pi \in \Pi^{(N)}$. For example, $L(\pi_{TD}^{(3)}) = \pi_{TD}^{(3)}$ and $L(\pi_{BU}^{(3)}) = \pi_{BU}^{(3)}$, although they are both in $\Pi_M^{(3)} \setminus \tilde{\Pi}^{(3)}$.

Remark. Furthermore, it is easy to check that $\tilde{\pi}_{IQ}^{(3)} = L(\pi_{IQ}^{(3)})$, since they differ only when $\zeta(t) = [1, 1, 1]$. In this case, $\tilde{\pi}_{IQ}^{(3)}$ schedules queues 1 and 3, while $\pi_{IQ}^{(3)}$ schedules Queue 2. But condition 1 ensures that $L(\pi_{IQ}^{(3)})$ chooses queues 1 and 3, whereby we get that $\tilde{\pi}_{IQ}^{(3)} = L(\pi_{IQ}^{(3)})$. Since $\pi_{IQ}^{(3)}$ is T.O., using Eqn. 17, we get, for all $t \geq 0$,

$$\begin{aligned} \sum_{i=1}^N Q_i^{\tilde{\pi}_{IQ}^{(3)}}(t) &\stackrel{st}{\leq} \sum_{i=1}^N Q_i^{\pi_{IQ}^{(3)}}, \\ \Rightarrow \sum_{i=1}^N \mathbb{E} Q_i^{\tilde{\pi}_{IQ}^{(3)}}(t) &\leq \sum_{i=1}^N \mathbb{E} Q_i^{\pi_{IQ}^{(3)}}, \end{aligned}$$

which immediately gives us

$$\begin{aligned} \limsup_{T \rightarrow \infty} \frac{1}{T} \sum_{t=0}^{T-1} \sum_{i=1}^N \mathbb{E}_{\tilde{\pi}_{IQ}^{(3)}} Q_i(t) \\ < \limsup_{T \rightarrow \infty} \frac{1}{T} \sum_{t=0}^{T-1} \sum_{i=1}^N \mathbb{E}_{\pi_{IQ}^{(3)}} Q_i(t) \\ < \infty, \end{aligned} \tag{20}$$

and shows that $\tilde{\pi}_{IQ}^{(3)}$ is T.O. as well. This provides another way to check the stability of $\tilde{\pi}_{IQ}^{(3)}$. This technique of producing a stable but *non MSM* policy and projecting it onto $\Pi_M^{(N)}$ is very important and will be used repeatedly in this paper to prove the throughput-optimality of many of the scheduling policies that we propose.

Remark. Eqn. (17) shows that $L(MW)$ is always strongly stable, since MW is. The proof proceeds along the same lines as the argument in Remark. V-C. In fact, the same proof can be easily generalized to show that If any policy $\pi \in \Pi^{(N)}$ is throughput-optimal, then so is $L(\pi)$.

Note that by our definition of the projection operator L and that of (see glossary of notation: XIV-B) the class $\Pi_M^{(N)}$, $L(MW)$ does *not* reside in this class, since the same occupancy vector $\zeta(t)$ can map to two different $s(t)$'s depending on what MW chooses in that slot. However, $L(MW)$ is an important policy and we will use it for comparison in our simulation results later.

Patently, all three policies we have proposed so far are MSM as they satisfy the 1st condition for a policy to be MSM (Sec. III) but only $\tilde{\pi}_{IQ}^{(3)}$ satisfies the 2nd condition as well. The projection operator L , as defined in [15], only results in an MSM policy and *not* an MSM policy that also prioritises inner queues. There are 4 MSM policies, of which only prioritises inner queues. If L was designed to map to MSM policies that also prioritise inner queues, then by Thm. 4.2 in [15] we could immediately conclude that in $\tilde{\Pi}^{(3)}$, the unique MSM policy that also prioritises inner queues, is delay optimal in $\Pi^{(3)}$. However,

L does ensure that the delay with $L(\pi)$ is no worse than that with $\pi \in \Pi^{(3)}$, and hence, we need to look for a uniformly delay optimal (see Defn. V-B) policy among the 4 MSM policies.

Now, since $\Pi_M^{(3)}$ also contains $\pi_{TD}^{(3)}$ and $\pi_{BU}^{(3)}$ and is *not* a singleton, it becomes necessary to examine the delay performance of the proposed policies in greater detail. It should be noted that the ideas used in this proof will form the basis for analysing the delay performance of policies for larger systems later.

Theorem 5. 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 $\tilde{\pi}_{IQ}^{(3)}$ be denoted by $\mathbf{Q}^{\tilde{\pi}_{IQ}^{(3)}}(t)$. Also let $\mathbf{Q}^\pi(0) = \mathbf{Q}^{\tilde{\pi}_{IQ}^{(3)}}(0)$. Then,

$$\sum_{i=1}^3 Q_i^{\tilde{\pi}_{IQ}^{(3)}}(t) \stackrel{st}{\leq} \sum_{i=1}^3 Q_i^\pi(t), \quad \forall t \geq 0, \quad (21)$$

where “ st ” denotes stochastic ordering.

The proof technique is essentially the same as that of Theorem 4.2 in [15], 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 $\tilde{\pi}_{IQ}^{(3)}$. The proof is deferred to Sec. XIV-F in the Appendix.

Directly analysing the stability and delay properties of the policies we propose in the sequel is very difficult. We therefore develop indirect methods to analyse them by first analysing non MSM 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.

D. Analysis of $\pi_{OQ}^{(3)}$

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

At time t :

- 1) If either $Q_1(t) > 0$ or $Q_3(t) > 0$, choose $(1, 0, 1)$.
- 2) Else choose $(0, 1, 0)$.

It turns out, analogous to the observation by McKeown et al [19] that this MSM policy is, in fact, not throughput-optimal.

Proposition 6 (MSM but not throughput-optimal). $\pi_{OQ}^{(3)}$ 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. XIV-G of the Appendix. Once again, this proof technique is important and we will repeatedly use it in the sequel.

This completes the characterization of $\Pi_M^{(3)}$.

E. Policies outside $\Pi_M^{(3)}$

We now propose and analyse a policy that we denote $\pi_{IQ}^{(3)}$, and show the rather surprising result that it is T.O. 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 simulation results in Sec. XII). 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 $Q_2(t) > 0$ choose $\mathbf{S}(t) = [0, 1, 0]$,
- 2) Else choose $\mathbf{s}(t) = [1, 0, 1]$.

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

Proposition 7 (A non-MSM but throughput-optimal policy). $\pi_{IQ}^{(3)}$ is throughput-optimal.

PROOF. The key tool behind the proof of this result is the throughput-optimality Lem. 2. It is easily checked that $\pi_{IQ}^{(3)}$ satisfies property \mathcal{P} in every slot and thus, by Lemma 2, is throughput-optimal. ■

VI. A RANDOMIZED POLICY: THE FLOW-IN-THE-MIDDLE PROBLEM

The “Flow-in-the-middle” problem, or FIM for short, is a fundamental problem faced by all networks that employ CSMA at the MAC layer. This problem has been studied in detail both analytically and experimentally in asynchronous *continuous-time* systems in the literature [26]–[31]. 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 $\zeta(t) := [\mathbb{I}_{\{Q_1(t)>0\}}, \mathbb{I}_{\{Q_2(t)>0\}}, \mathbb{I}_{\{Q_3(t)>0\}}]$. Consider the policy $\rho_\gamma^{(3)}$ indexed⁷ by a randomization parameter $\gamma \in [0, 1]$ defined as follows.

At time t :

- If $\zeta(t) = [1, 1, 1]$ or $[1, 0, 1]$, then $\mathbf{S}(t) = [1, 0, 1]$.
- Else, if $\zeta(t) = [1, 1, 0]$ 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, $\mathbf{S}(t) = \zeta(t)$.

Clearly, this policy is a randomization between the two 3-queue MSM policies $\tilde{\pi}_{IQ}^{(3)}$ and $\pi_{OQ}^{(3)}$. A comparison of the definitions of $\rho_\gamma^{(3)}$, $\tilde{\pi}_{IQ}^{(3)}$ and $\pi_{OQ}^{(3)}$ clearly shows that $\rho_\gamma^{(3)}$ essentially chooses $\tilde{\pi}_{IQ}^{(3)}$ w.p. γ and $\pi_{OQ}^{(3)}$ w.p. $1 - \gamma$.

A. Analysis of $\rho_\gamma^{(3)}$

Proposition 8. $\rho_\gamma^{(3)}$ is unstable for $\gamma \in [0, 0.5]$.

PROOF. The basic idea for proving instability is the same as the one we used to prove that $\pi_{OQ}^{(3)}$ is unstable (Prop. 6). For every $\gamma \in [0, 0.5]$ we show the existence of an arrival rate vector sufficiently close to the boundary of the capacity region such that the policy is not able to stabilize it.

Towards that end, consider the processes embedded at instant t (we will drop the time index, i.e. t , for simplicity of notation)

$$\begin{aligned}
 P\{S_2 = 1\} &= P\left\{S_2 = 1 \mid Q_1 + Q_3 = 0\right\} P\{Q_1 + Q_3 = 0\} \\
 &+ P\left\{S_2 = 1 \mid Q_1 > 0, Q_3 = 0\right\} P\{Q_1 > 0, Q_3 = 0\} \\
 &+ P\left\{S_2 = 1 \mid Q_1 = 0, Q_3 > 0\right\} P\{Q_1 = 0, Q_3 > 0\} \\
 &+ P\left\{S_2 = 1 \mid Q_1 > 0, Q_3 > 0\right\} P\{Q_1 > 0, Q_3 > 0\} \\
 &= 1 \cdot P\{Q_1 + Q_3 = 0\} + \gamma \cdot P\{Q_1 > 0, Q_3 = 0\} + \gamma \cdot P\{Q_1 = 0, Q_3 > 0\} \\
 &+ 0 \cdot P\{Q_1 > 0, Q_3 > 0\} \\
 &= \gamma(1 - P\{Q_1 > 0, Q_3 > 0\}) + (1 - \gamma)P\{Q_1 = 0, Q_3 = 0\}.
 \end{aligned}$$

Now, notice that $\{A_1 > 0, A_3 > 0\} \subset \{Q_1 > 0, Q_3 > 0\}$ which means that $P\{A_1 > 0, A_3 > 0\} = \lambda_1 \lambda_3 \leq P\{Q_1 > 0, Q_3 > 0\}$. Similarly, $(1 - \lambda_1)(1 - \lambda_3) \geq P\{Q_1 = 0, Q_3 = 0\}$. Using this in the above equation, we get

$$\begin{aligned}
 P\{S_2 = 1\} &\leq \gamma \cdot (1 - \lambda_1 \lambda_3) + (1 - \gamma) \cdot (1 - \lambda_1)(1 - \lambda_3) \\
 &= 1 - (1 - \gamma)(\lambda_1 + \lambda_3) + (1 - 2\gamma)\lambda_1 \lambda_3.
 \end{aligned}$$

Consider the arrival rate vector $\lambda = \lambda \cdot [1, 1, 1]$, where, obviously, $\lambda < 0.5$. For this vector,

$$P\{S_2 = 1\} \leq (1 - 2\gamma)\lambda^2 - 2(1 - \gamma)\lambda + 1.$$

Therefore, if we can show the existence of some $\lambda \in [0, 0.5]$ such that $(1 - 2\gamma)\lambda^2 - 2(1 - \gamma)\lambda + 1 < \lambda$, we will be done. Consider the polynomial $p(\lambda) := (1 - 2\gamma)\lambda^2 - (3 - 2\gamma)\lambda + 1$. $p(0) = 1 > 0$, but

$$\begin{aligned}
 p(0.5) &= \frac{1 - 2\gamma}{4} - \frac{3 - 2\gamma}{2} + 1 \\
 &= -\frac{5}{4} + \frac{\gamma}{2} + 1 \\
 &= -\frac{1}{4} + \frac{\gamma}{2} \stackrel{*1}{<} 0,
 \end{aligned} \tag{22}$$

where, in *1 we have used the fact that $\gamma \in [0, 0.5]$. By continuity, therefore, there exists some $\lambda \in [0, 0.5]$ for every such γ , such that Queue 2 cannot be stabilized. ■

Now, consider the set of arrival rates

$$\Lambda_\gamma^{(3)} := \left\{ \lambda \in \mathbb{R}_+^3 \mid \lambda_1 + \lambda_2 < \gamma, \lambda_2 + \lambda_3 < \gamma \right\} \tag{23}$$

Proposition 9. For every $\gamma \in (0, 1]$, $\rho_\gamma^{(3)}$ stabilizes all rate vectors in $\Lambda_\gamma^{(3)}$.

⁷We use ρ instead of π to highlight the fact that this is a *randomized* policy.

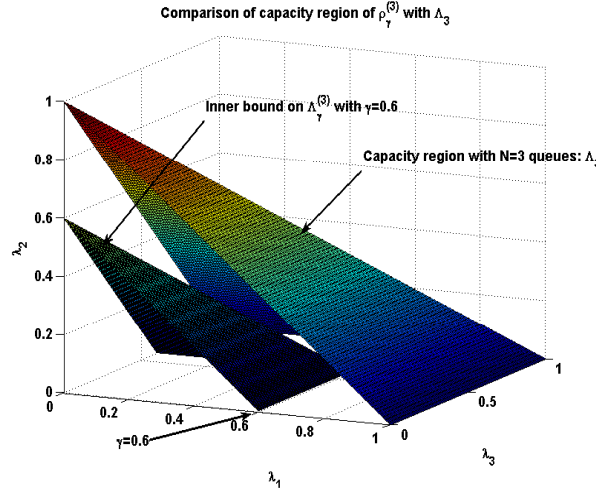


Fig. 8: Figure showing the inner bound on the stability region of $\rho_\gamma^{(3)}$ given in (23). The region below the outer surface, towards the origin, in the positive orthant is the capacity region Λ_3 .

Remark. The above result means that the stability region of $\rho_\gamma^{(3)} \nearrow \Lambda_3$ as $\gamma \uparrow 1$, i.e., for all $0 < \gamma_1 \leq \gamma_2 \leq 1$, $\Lambda_{\gamma_1}^{(3)} \subseteq \Lambda_{\gamma_2}^{(3)}$. This is consistent with the fact that $\rho_\gamma^{(3)} \Big|_{\gamma=1} \equiv \tilde{\pi}_{IQ}^{(3)}$.

PROOF. Recall that $D_i(t)$ is the actual number of packets leaving Queue i at the end of time slot t . We first focus our attention on the pair of queues 1 and 2. Now, from the definition of $\rho_\gamma^{(3)}$ we see that this pair receives service whenever

- $\zeta_1(t) = 1$, i.e., when $\zeta(t)$ is $[1, 0, 0]$, $[1, 0, 1]$, $[1, 1, 0]$ or $[1, 1, 1]$, w.p. I ,
- $\zeta(t) = [0, 1, 1]$, w.p. γ , and
- $\zeta(t) = [0, 1, 0]$, w.p. I .

Once again, we drop the time index for simplicity of notation. From the above discussion, we see that

$$\begin{aligned}
 \mathbb{E} \left[D_1 + D_2 \middle| \mathbf{Q} \right] &= 1 \cdot \mathbb{I}_{\{Q_1 > 0\}} + \gamma \cdot \mathbb{I}_{\{Q_1=0, Q_2 > 0, Q_3 > 0\}} \\
 &\quad + 1 \cdot \mathbb{I}_{\{Q_1=0, Q_2 > 0, Q_3=0\}} \\
 &= (\gamma + 1 - \gamma) \cdot \left(\mathbb{I}_{\{Q_1 > 0, Q_2 > 0\}} + \mathbb{I}_{\{Q_1 > 0, Q_2=0\}} \right) \\
 &\quad + \gamma \cdot \mathbb{I}_{\{Q_1=0, Q_2 > 0, Q_3 > 0\}} \\
 &\quad + (\gamma + 1 - \gamma) \cdot \mathbb{I}_{\{Q_1=0, Q_2 > 0, Q_3=0\}} \\
 &= \gamma \left[\mathbb{I}_{\{Q_1 > 0, Q_2 > 0\}} + \mathbb{I}_{\{Q_1 > 0, Q_2=0\}} + \mathbb{I}_{\{Q_1=0, Q_2 > 0\}} \right] \\
 &\quad + (1 - \gamma) \left[\mathbb{I}_{\{Q_1 > 0\}} + \mathbb{I}_{\{Q_1=0, Q_2 > 0, Q_3=0\}} \right] \\
 &= \gamma \cdot \mathbb{I}_{\{Q_1+Q_2 > 0\}} + (1 - \gamma) \left[\mathbb{I}_{\{Q_1 > 0\}} + \mathbb{I}_{\{Q_1=0, Q_2 > 0, Q_3=0\}} \right] \\
 &\geq \gamma \cdot \mathbb{I}_{\{Q_1+Q_2 > 0\}}.
 \end{aligned}$$

Using the same procedure as above, we also get $\mathbb{E} \left[D_2 + D_3 \middle| \mathbf{Q} \right] \geq \gamma \cdot \mathbb{I}_{\{Q_2+Q_3 > 0\}}$. Now, we are in a position to invoke a γ randomized version of our Property \mathcal{P} , as follows. Let $L(\mathbf{Q}(t)) := (Q_1(t) + Q_2(t))^2 + (Q_2(t) + Q_3(t))^2$ be the Lyapunov function.

We focus on the first term only; the analysis of the second term follows the same procedure.

$$\begin{aligned}
& (Q_1(t+1) + Q_2(t+1))^2 - (Q_1(t) + Q_2(t))^2 \\
&= (Q_1(t) - D_1(t) + A_1(t+1) + Q_2(t) - D_2(t) \\
&\quad + A_2(t+1))^2 - (Q_1(t) + Q_2(t))^2 \\
&\stackrel{*2}{\leq} 1 + 4 - 2(Q_1(t) + Q_2(t))(D_1(t) + D_2(t) \\
&\quad - A_1(t+1) - A_2(t+1))
\end{aligned}$$

In *2, we have used several facts. For any 3 non negative reals, x, y, z , $(x - y + z)^2 \leq x^2 + y^2 + z^2 - 2x(y - z)$. Furthermore, $D_i(t+1) \leq 1$, $\forall t$ and $(A_1(t+1) + A_2(t+1))^2 \leq 4$, $\forall t$. So,

$$\begin{aligned}
& \mathbb{E} \left[(Q_1(t+1) + Q_2(t+1))^2 - (Q_1(t) + Q_2(t))^2 \middle| \mathbf{Q}(t) = \mathbf{Q} \right] \\
&\leq 5 - 2(Q_1(t) + Q_2(t)) \left(\mathbb{E} [D_1(t) + D_2(t) \middle| \mathbf{Q}] - (\lambda_1 + \lambda_2) \right) \\
&\stackrel{*3}{\leq} 5 - 2(Q_1(t) + Q_2(t)) \left(\gamma \cdot \mathbb{I}_{\{Q_1+Q_2>0\}} - (\lambda_1 + \lambda_2) \right).
\end{aligned}$$

We now use the fact that for any non negative random variable X , $\mathbb{E}X\mathbb{I}_{\{X>0\}} = \mathbb{E}X$.

$$\begin{aligned}
& \mathbb{E} \left[(Q_1(t+1) + Q_2(t+1))^2 - (Q_1(t) + Q_2(t))^2 \right] \\
&\leq 5 - 2\mathbb{E}(Q_1(t) + Q_2(t))(\gamma - (\lambda_1 + \lambda_2)). \\
&\stackrel{*4}{\leq} 5 - 2\epsilon_{1,2}\mathbb{E}(Q_1(t) + Q_2(t)),
\end{aligned}$$

where in *4 we have used the fact that $\lambda \in \Lambda_\gamma^{(3)}$ to conclude that $(\gamma - (\lambda_1 + \lambda_2)) =: \epsilon_{1,2} > 0$. The proof of strong stability follows by showing the existence of an $\epsilon_{2,3}$ for the queues 2 and 3 and putting together both these upper bounds to get an upper bound on the Lyapunov function and using the *telescoping sum* argument.

■

VII. PATH GRAPH CONFLICT MODELS WITH $N > 3$: POLICY SPLICING FOR THROUGHPUT OPTIMAL QNB SCHEDULING

The previous section was devoted to introducing the reader to the idea of scheduling policies that rely only on occupancy information (empty-nonempty status of queues) and examining 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.

The path we shall follow uses a “policy splicing” technique to construct MSM policies for large systems by splicing together MSM policies for smaller systems. We first give a high-level overview of the technique. Recall the “Top-Down” and “Bottom-Up” policies, $\pi_{TD}^{(3)}$ and $\pi_{BU}^{(3)}$, discussed in Sec. V-A. For a general path-graph network with N queues (communication links), the “Top-Down” policy, $\pi_{TD}^{(N)}$, which maps an occupancy vector $\zeta(t)$ 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 $\mathbf{Q}(t) = [Q_1(t), \dots, Q_N(t)]$ is the queue length vector at time t , then the **occupancy vector** at time t is defined by $\zeta(t) = [\mathbb{I}_{\{Q_1(t)>0\}}, \dots, \mathbb{I}_{\{Q_N(t)>0\}}]$.

At time t

1) For $j=1:N$

- a) If $\zeta_j(t) = 1$ and $s_{j-1}(t) = 0$, then $s_j(t) = 1$ and $s_{j+1}(t) = 0$.
- b) Else if $\zeta_j(t) = 1$ and $s_{j-1}(t) = 1$, then $s_j(t) = 0$.
- c) 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 immediately.

Proposition 10. $\pi_{TD}^{(N)}$ and $\pi_{BU}^{(N)}$ are MSM for all $N \in \mathbb{N}$.

PROOF. Lem. 1 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.

Recall that in Lem. 1, given an occupancy vector ζ , we defined $k = k(\zeta)$ to be the twice the number of runs of nonempty queues. Also, $j_1 = j_1(\zeta), \dots, j_k = j_k(\zeta)$, were defined to be the nonempty queues that mark the beginnings ($j_{\{\text{odd subscript}\}}$) and ends ($j_{\{\text{even subscript}\}}$) of the nonempty runs, or are the two outermost queues. We 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 1c 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 1a. Thereafter, since all queues between Queues j_{2m-1} and j_{2m} (including these two) are nonempty, the policy alternates between Conditions 1a and 1b, scheduling every alternate queue and thus satisfying Condition 1 in Lem. 1.
- 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. 1.

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

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. Consider a system of $2N - 1$ queues, $N \geq 1$. In Algorithm 1, we splice the TD and BU policies and construct a scheduling policy⁸ $\pi_{SP}^{(2N-1)}$ on this system. **Note:** We assume the presence of the two *virtual queues* Queue 0 and Queue $2N$ here as well. Before proceeding to analyse this policy, we first need to make sure it really is a well-defined policy, i.e., it

Data: Binary occupancy vector $\zeta(t)$

Result: 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. 9.

provides a valid activation vector for each of the 2^{2N-1} possible occupancy vectors.

Lemma 11. $\pi_{SP}^{(2N-1)}$, as defined above, is well-defined.

PROOF. See Sec. XIV-H in the Appendix. ■

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 $\{N, N + 1, \dots, 2N - 1\}$ and $\{1, 2, \dots, N\}$ respectively. The following result follows from the definition of the splicing process.

⁸The subscript “SP” refers to the fact that this is a Spliced Policy.

Recall from Sec. II-B that the capacity region of a path-graph interference network consisting of N queues is defined by

$$\Lambda_N := \left\{ \lambda \in \mathbb{R}_+^N \mid \lambda_i + \lambda_{i+1} < 1, \ 1 \leq i \leq N-1 \right\}, N \in \mathbb{N}. \quad (24)$$

Theorem 12. 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. 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 $\lambda \in \Lambda_N$,

$$\limsup_{T \rightarrow \infty} \frac{1}{T} \sum_{t=0}^{t-1} \sum_{i=N}^{2N-1} \mathbb{E}_{\pi_{TD}^{(N)}} Q_i(t) < \infty, \text{ and} \quad (25)$$

$$\limsup_{T \rightarrow \infty} \frac{1}{T} \sum_{t=0}^{t-1} \sum_{i=1}^N \mathbb{E}_{\pi_{BU}^{(N)}} Q_i(t) < \infty. \quad (26)$$

Notice in particular the indices of the inner summations in the above inequalities. Now, for every $\lambda \in \Lambda_{2N-1}$ define $\lambda_{1:N} = [\lambda_1, \dots, \lambda_N]$ and $\lambda_{N:2N-1} = [\lambda_N, \dots, \lambda_{2N-1}]$ and notice that by the definition of Λ_N , $\lambda_{1:N} \in \Lambda_N$ and $\lambda_{N:2N-1} \in \Lambda_N$, which means that (25) and (26) are still separately true. The proof concludes when we observe that

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

■

Remark. 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 $\zeta_j(t) = 0, \forall j \in \{1, \dots, N-2\} \cup \{N+2, \dots, 2N-1\}$, 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.

To reduce delay, one needs to extract an MSM policy from this spliced policy. Fortunately, the procedure to accomplish this has already been described in Sec. V-C. We simply project the policy onto the space $\Pi_M^{(N)}$ of MSM policies using the projection operator defined therein. Thereafter, we use some observations based on Condition 3 in Lem. 1 to improve the delay performance of this projected MSM policy by finally obtaining a policy in $\tilde{\Pi}^{(N)}$. Figures 9 and 10 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 $1 \leq k \leq 2N$. All one needs is 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 9 and 10)

a) General splicing procedure:

- 1) Proposing TD and 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 the resulting policy to break all ties in favor of inner queues to get a policy with better delay performance than the MSM policy.

We have already proposed and analysed 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, we show how low delay policies for $2 \times 3 - 1 = 5$ -queue systems can be constructed from these two 3-queue policies.

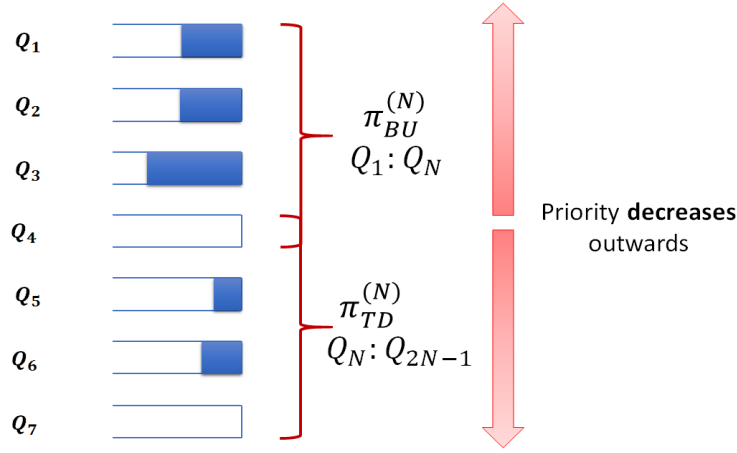


Fig. 9: Illustrating the manner in which the policies $\pi_{TD}^{(N)}$ and $\pi_{BU}^{(N)}$ are spliced together to form the *non-MSM* policy $\pi_{SP}^{(2N-1)}$. Note that the splicing is consistent in that even though the two sub-policies schedule over overlapping sections of queues, their decisions do not contradict each other.

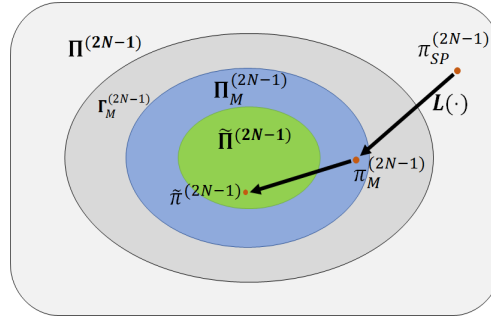


Fig. 10: Illustrating steps 3 and 4 of the general policy splicing process. The queue nonemptiness-based, non-MSM policy $\pi_{SP}^{(2N-1)}$ is first projected into $\Pi_M^{(N)}$ to get the *MSM* policy $\pi_M^{(2N-1)} \equiv L(\pi_{SP}^{(2N-1)})$. Thereafter, $\pi_M^{(2N-1)}$ is modified to prioritize inner queues to get $\tilde{\pi}^{(2N-1)} \in \tilde{\Pi}^{(2N-1)}$.

A. Low-delay Scheduling Policies for Systems with $N = 5$ queues

Our goal is now to begin with the two 3-queue scheduling policies $\pi_{TD}^{(3)}$ and $\pi_{BU}^{(3)}$ and use the general policy splicing procedure VII-0a to design a throughput optimal 5-queue scheduling policy $\tilde{\pi}^{(5)}$, in $\tilde{\Pi}^{(5)}$. The purpose of focusing on $\tilde{\Pi}^{(5)}$ is, as stated before, to ensure that the policy (if one such exists) will show desirable delay properties. We will begin with a non delay optimal policy, which we call $\pi_M^{(5)}$, since it is *MSM*. The stability of $\pi_M^{(5)}$ will be proved using a non *MSM* policy, $\pi_{SP}^{(5)}$, and finally, we will show that $\tilde{\pi}^{(5)}$ is throughput optimal by invoking certain properties of policies in $\tilde{\Pi}^{(5)}$.

1) *Analysis of $\tilde{\pi}^{(5)}$* : The spliced policy $\pi_{SP}^{(5)}$, is defined as in the definition of the general spliced policy (Defn. 1) and is throughput optimal as shown in Thm. 12, since $\pi_{TD}^{(3)}$ and $\pi_{BU}^{(3)}$ have both been shown to be throughput optimal in Sec. V-A. Since the projection operator L preserves strong stability, the *MSM* policy $\pi_M^{(5)} \equiv L(\pi_{SP}^{(5)})$, which lies in $\Pi_M^{(5)}$, is also throughput optimal. The *MSM* policy $\pi_M^{(5)}$ is defined as follows

At time t :

- 1) If $\zeta(t) = [0, 1, 1, 1, 0]$, $\mathbf{s}(t) = [0, 1, 0, 1, 0]$
- 2) Else, if $Q_3(t) > 0$, $\mathbf{s}(t) = [1, 0, 1, 0, 1]$.
- 3) Else, if $Q_2(t) > 0$ and $Q_4(t) > 0$, $\mathbf{s}(t) = [0, 1, 0, 1, 0]$.
- 4) Else, if $Q_2(t) > 0$ or $Q_4(t) > 0$,
 - a) $\mathbf{s}(t) = [0, 1, 0, 0, 1]$ if $Q_2(t) > 0$ and
 - b) $\mathbf{s}(t) = [1, 0, 0, 1, 0]$ if $Q_4(t) > 0$.
- 5) Else, $\mathbf{s}(t) = [1, 0, 1, 0, 1]$.

In an effort to improve the delay performance of the above policy, we modify $\pi_M^{(5)}$ to prioritise inner queues (here, Queues 2, 3 and 4). We accomplish this by using the conditions specified in Lem. 1 that characterises *MSM* policies on path graph networks. The first two conditions by themselves form a necessary and sufficient condition for a policy to be *MSM*, and it is easy to show that $\pi_M^{(5)}$ satisfies both. To reduce delay, however, one can use Condition 3 therein that helps prioritise inner queues. Consider the definition of $\pi_M^{(5)}$ once more, and notice that under this policy,

- $\zeta(t) = [1, 1, 1, 1, 0] \mapsto \mathbf{s}(t) = [1, 0, 1, 0, 1]$, and

- $\zeta(t) = [0, 1, 1, 1, 1] \mapsto \mathbf{s}(t) = [1, 0, 1, 0, 1]$.

Going back to the discussion on MSM policies, Condition 3 in Lem. 1 states that when $j_1 = 1$,

$$S_j(t) = \begin{cases} 1 & \text{if } j_2 - j \text{ is even, } j_1 \leq j \leq j_2, \\ 0 & \text{otherwise,} \end{cases}$$

and similarly for the case with $j_2 = N$. This means when $\mathbf{s}(t) = [1, 1, 1, 1, 0]$, since $j_1 = 1$, and $j_2 = 4$, we really should have $\mathbf{s}(t) = [0, 1, 0, 1, 0]$. Similarly, when $\zeta(t) = [0, 1, 1, 1, 1]$, $j_1 = 2$ and $j_2 = 5 = N$, and using Condition 3 once again, we see that $\mathbf{s}(t) = [0, 1, 0, 1, 0]$. Including these two corrections in the definition of $\pi_M^{(5)}$ gives rise to the following policy, that we call $\tilde{\pi}^{(5)}$.

At time t :

- 1) If $\zeta(t) = [0, 1, 1, 1, 0]$, $\mathbf{s}(t) = [0, 1, 0, 1, 0]$.
- 2) If $\zeta(t) = [1, 1, 1, 1, 0]$ or $[0, 1, 1, 1, 1]$, $\mathbf{s}(t) = [0, 1, 0, 1, 0]$.
- 3) Else, if $Q_3(t) > 0$, $\mathbf{s}(t) = [1, 0, 1, 0, 1]$.
- 4) Else, if $Q_2(t) > 0$ and $Q_4(t) > 0$, $\mathbf{s}(t) = [0, 1, 0, 1, 0]$.
- 5) Else, if $Q_2(t) > 0$ or $Q_4(t) > 0$,
 - a) $\mathbf{s}(t) = [0, 1, 0, 0, 1]$ if $Q_2(t) > 0$ and
 - b) $\mathbf{s}(t) = [1, 0, 0, 1, 0]$ if $Q_4(t) > 0$.
- 6) Else, $\mathbf{s}(t) = [1, 0, 1, 0, 1]$.

Clearly, $\tilde{\pi}^{(5)}$ satisfies all three requirements of Lem. 1 and hence, resides in $\tilde{\Pi}^{(5)}$. As stated before, the objective is to show that $\tilde{\pi}^{(5)} \in \tilde{\Pi}^{(5)}$ gives rise to stochastically smaller sum queue lengths than $\pi_M^{(5)} \in \Pi_M^{(5)}$. This, however, is quite obvious from the prior discussion in this section since $\tilde{\pi}^{(5)}$ always satisfies Condition 3 in Lem. 1. Specifically, when $\zeta(t) = [1, 1, 1, 1, 0]$ or $[0, 1, 1, 1, 1]$, $\pi_M^{(5)}$ chooses $\mathbf{s}(t) = [1, 0, 1, 0, 1]$, while $\tilde{\pi}^{(5)}$ chooses $\mathbf{s}(t) = [0, 1, 0, 1, 0]$. Using the same technique as in the proof of Prop. 20, we see that for any arrival rate $\lambda \in \Lambda_S^o$,

$$\sum_{i=1}^5 Q_i^{\tilde{\pi}^{(5)}}(t) \stackrel{st}{\leq} \sum_{i=1}^5 Q_i^{\pi_M^{(5)}}(t), \quad \forall t \geq 0.$$

In fact, such a refinement of every policy in $\Pi_M^{(5)} \setminus \tilde{\Pi}^{(5)}$ can be obtained in a similar manner by enforcing Condition 3 in Lem. 1 to reduce delay. With this, we see that.

Proposition 13. For every policy $\pi \in \Pi_M^{(5)} \setminus \tilde{\Pi}^{(5)}$, there exists a policy $\pi' \in \tilde{\Pi}^{(5)}$ such that

$$\sum_{i=1}^5 Q_i^{\pi'}(t) \stackrel{st}{\leq} \sum_{i=1}^5 Q_i^{\pi}(t), \quad \forall t \geq 0. \quad (27)$$

To summarize, in this section, we first proposed a general algorithm 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 also provided an explicit example of the entire process using two of the policies already studied ($\pi_{TD}^{(3)}$ and $\pi_{BU}^{(3)}$). We now move on to proposing and analysing top-down and bottom-up priority policies for larger systems.

VIII. TOP-DOWN AND BOTTOM-UP POLICIES FOR SYSTEMS WITH $N = 4$ AND 5 QUEUES

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 $N = 2 \times 4 - 1 = 7$ 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) Else⁹, $\mathbf{s}(t) = (1, 0, 0, 1)$.
- 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 14. $\pi_{TD}^{(4)}$ is throughput optimal.

⁹Strictly 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 $\zeta_4(t) = 0$ doesn't violate interference constraints since it means that Queue 4 is empty.

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. XIV-I in the Appendix. ■

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 throughput optimal using similar arguments. We then define $\pi_{SP}^{(7)}$ as in 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. 12, 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 VII-0a.

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

Proposition 15. $\pi_{TD}^{(5)}$ is throughput optimal.

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

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

IX. PATH-GRAPH CONFLICT MODELS WITH $N > 3$: DELAY WITH QNB POLICIES

We now turn our attention to the vital aspect of delay. We have already proved, in Thm. 5, that for the system with $N = 3$ queues, there exists a unique uniformly delay-optimal policy, that we named $\tilde{\pi}_{IQ}^{(3)}$. 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*.

General Flow of the Section: We begin with a study of the class $\tilde{\Pi}^{(4)}$ of policies for systems with $N = 4$ queues. This class contains policies that are both MSM and prioritize inner queues (here, Queues 2 and 3), i.e, they satisfy all three conditions in Lem. 1. The number of policies in this class, $|\tilde{\Pi}^{(4)}|$, is 4 and we show that all four are throughput optimal. Thereafter, we show that for every policy in $\Pi_M^{(4)}$, the class of MSM queue nonemptiness-based policies, there exists a policy in $\tilde{\Pi}^{(4)}$ that shows better delay performance. However, we finally show that different policies in $\tilde{\Pi}^{(4)}$ perform better than the others (in terms of delay) over different portions of the capacity region Λ_4 , which means that there is no *uniformly delay optimal* (see glossary XIV-B) policy for 4-queue systems. The general proof then follows from a contradiction argument.

A. Characterizing the Class $\tilde{\Pi}^{(4)}$

In Sec. V we saw how the MSM property helped whittle down the number of queue nonemptiness-based policies in $\Pi^{(3)}$, making the final analysis of the class $\Pi_M^{(3)}$ tractable. Continuing along the same lines, with 4 queues, we have three activation vectors to choose from in any given slot, viz., $[1, 0, 1, 0]$, $[1, 0, 0, 1]$ and $[0, 1, 0, 1]$, and $\zeta(t)$ can take one of $2^4 = 16$ values. The number of queue nonemptiness-based policies within $\Pi^{(4)}$ is, therefore, $3^{2^4} > 43 \times 10^6$. However, we must keep in mind that we will be dealing with MSM policies that serve the largest number of noninterfering queues in every slot.

Observe Columns 1 and 2 in Table V. The second gives all possible activation vectors an MSM policy can choose from for every occupancy vector $\zeta(t)$ (the choice with $\zeta(t) = 0$ is inconsequential since it refers to when the system is empty). Most have a single activation vector associated with them meaning MSM policies don't have any choice for such occupancies, but there are 5 vectors each with 2 valid activation vectors and 1 with 3 valid activation vectors, giving rise to $25 \times 3 = 96$ policies. So, the number of policies gets whittled down to 96 policies once the MSM condition is imposed. But since this number is also inordinately large, we will restrict our study to $\tilde{\Pi}^{(4)}$, which contains 4 policies as shown in Column 3 in Table V. We denote them by $\{\tilde{\pi}_i^{(4)}, 1 \leq i \leq 4\}$.

In what follows, we provide a complete characterization of $\tilde{\Pi}^{(4)}$. We first show that the policies $\tilde{\pi}_1^{(4)}$ and $\tilde{\pi}_2^{(4)}$ are stable using the policy splicing technique on some of the 3-queue policies whose stability we have already established. Then, we prove the stability of the other two policies using an extension of the stochastic ordering technique we used to show the delay optimality of $\tilde{\pi}_{IQ}^{(3)}$ in Thm. 5.

1) *Analysis of $\tilde{\pi}_1^{(4)}$ and $\tilde{\pi}_2^{(4)}$:* We define the policy $\tilde{\pi}_1^{(4)}$ as follows.

At time t ,

- 1) If $\zeta(t) = [1, 1, 1, 0]$, $\mathbf{s}(t) = [1, 0, 1, 0]$.
- 2) Else, if $Q_2(t) > 0$, $\mathbf{s}(t) = [0, 1, 0, 1]$.
- 3) Else, if $Q_3(t) > 0$, $\mathbf{s}(t) = [1, 0, 1, 0]$.
- 4) Else, $\mathbf{s}(t) = [1, 0, 0, 1]$.

The policy $\tilde{\pi}_2^{(4)}$ simply swaps the priorities of Queues 2 and 3. For clarity and completeness, we define $\tilde{\pi}_2^{(4)}$ explicitly below. At time t ,

- 1) If $\zeta(t) = [0, 1, 1, 1]$, $\mathbf{s}(t) = [0, 1, 0, 1]$.
- 2) Else, if $Q_3(t) > 0$, $\mathbf{s}(t) = [1, 0, 1, 0]$.
- 3) Else, if $Q_2(t) > 0$, $\mathbf{s}(t) = [0, 1, 0, 1]$.
- 4) Else, $\mathbf{s}(t) = [1, 0, 0, 1]$.

Table. II provides the complete enumeration of $\tilde{\pi}_1^{(4)}$, i.e., for every occupancy vector $\zeta(t)$.

Remark. Firstly, the presence of a default, i.e., condition 4, ensures that the above definitions are exhaustive, in the sense that they cover all $2^4 = 16$ states of $\zeta(t)$. This means, using the terminology in Sec. VIII, that both these policies are *admissible*. This will become important when we restate the same policies in a different manner while proving their stability.

Remark. We next note that while these policies have been proposed in [16], only an informal argument regarding their stability properties has been provided therein, followed by a study of their performance in the Halfin Whitt regime. The informal argument therein asserts that the fraction of time for which Queue 2 is nonempty equals its arrival rate λ_2 , and this claim is crucial to their stability argument.

By Little's Theorem applied to the HOL position, this assertion holds only if the mean waiting time in the HOL position of Queue 2 is exactly 1 slot. The actual fraction of time for which Queue 2 is nonempty converges to $\lambda_2 \cdot \mathbb{E}B_2$, where $\mathbb{E}B_2$ is the mean service time of packets in Queue 2. Since Queue 2 is not always served whenever it is nonempty, $\mathbb{E}B_2 > 1$ (*strict inequality*), so the fraction of time left to offer service to Queue 1 is strictly smaller than $1 - \lambda_2$. Moreover, as can be seen from the definition of $\tilde{\pi}_1^{(4)}$, $\mathbf{s}(t)$ is not necessarily $[1, 0, 1, 0]$ whenever Queues 1 and 3 are nonempty (take $\zeta(t) = [1, 1, 1, 1]$, for instance). It is, therefore, unclear whether Queue 1 is offered service often enough to stabilize it. We here provide a formal proof of the throughput optimality of $\tilde{\pi}_1^{(4)}$ and $\tilde{\pi}_2^{(4)}$.

Hence, the fact that this policy is stabilizing requires an actual argument. We here provide a formal proof of the throughput optimality of $\tilde{\pi}_1^{(4)}$ and $\tilde{\pi}_2^{(4)}$. In the process, we derive another throughput optimal policy which is *not* MSM and complete the proof using a stochastic dominance argument.

Proposition 16. $\tilde{\pi}_1^{(4)}$ is throughput optimal.

PROOF. Before proceeding with the proof, notice that the definition of $\tilde{\pi}_1^{(4)}$ can be restated as follows.

At time t :

- 1) If $\zeta(t) = [1, 1, 1, 0]$, $\mathbf{s}(t) = [1, 0, 1, 0]$.
- 2) Else,
 - a) Check $[\zeta_1(t), \zeta_2(t), \zeta_3(t)]$
 - i) If $\zeta_2(t) = 1$, $\mathbf{s}(t) = [0, 1, 0, 1]$.
 - ii) Else if $\zeta_3(t) = 1$, $\mathbf{s}(t) = [1, 0, 1, 0]$.
 - iii) Else $\mathbf{s}(t) = [1, 0, 0, 1]$.
 - b) Check $[\zeta_2(t), \zeta_3(t), \zeta_4(t)]$
 - i) If $\zeta_2(t) = 1$, $\mathbf{s}(t) = [0, 1, 0, 1]$.
 - ii) Else if $\zeta_3(t) = 1$, $\mathbf{s}(t) = [1, 0, 1, 0]$.
 - iii) Else $\mathbf{s}(t) = [1, 0, 0, 1]$.

Clearly, for a given $\zeta(t)$ the actions recommended in 2a and 2b above never contradict each other despite depending on overlapping subsets of queues. From the above definition, if condition 1, i.e., the check for $\zeta(t) = [1, 1, 1, 0]$ is removed, the resulting policy, which we call $\pi_{TI}^{(4)}$, is no longer MSM. Table. II shows the difference between the two policies. It should be noted that the policy $\pi_{TI}^{(4)}$ is also constructed by splicing together two 3-queue policies. However, since the subscript “SP” has been reserved for policies constructed by only splicing $\pi_{TD}^{(N)}$ and $\pi_{BU}^{(N)}$, and since the constituent policies of $\pi_{TI}^{(4)}$ are *not* the Top-Down Bottom-Up pair, we have used a different subscript. We now state the following result.

Lemma 17. $\pi_{TI}^{(4)}$ is Throughput Optimal.

PROOF. See Sec. XIV-L in the Appendix. ■

Recall the projection operator $L : \Pi^{(N)} \rightarrow \Gamma_M^{(N)}$ defined in Sec. V-C. It can now be observed that $\tilde{\pi}_1^{(4)} = L(\pi_{TI}^{(4)})$, since the only occupancy vector for which $\pi_{TI}^{(4)}$ does not serve the maximum number of queues is $[1, 1, 1, 0]$. When this shortcoming is rectified, we get $\tilde{\pi}_1^{(4)}$. Now, we use the discussion in Sec. V, specifically, Eqn. (17), to see that

$$\sum_{i=1}^4 Q_i^{\tilde{\pi}_1^{(4)}}(t) \leq \sum_{i=1}^4 Q_i^{\pi_{TI}^{(4)}}(t), \quad \forall t \geq 0. \quad (28)$$

Finally, since $\pi_{TI}^{(4)}$ is strongly stable, so is $\tilde{\pi}_1^{(4)}$ as shown below

$$\begin{aligned} \sum_{i=1}^4 Q_i^{\tilde{\pi}_1^{(4)}}(t) &\leq \sum_{i=1}^4 Q_i^{\pi_{TI}^{(4)}}, \\ \Rightarrow \sum_{i=1}^4 \mathbb{E} Q_i^{\tilde{\pi}_1^{(4)}}(t) &\leq \sum_{i=1}^4 \mathbb{E} Q_i^{\pi_{TI}^{(4)}}, \end{aligned}$$

which immediately gives us

$$\begin{aligned} \limsup_{T \rightarrow \infty} \frac{1}{T} \sum_{t=0}^{T-1} \sum_{i=1}^4 \mathbb{E}_{\tilde{\pi}_1^{(4)}} Q_i(t) \\ < \limsup_{T \rightarrow \infty} \frac{1}{T} \sum_{t=0}^{T-1} \sum_{i=1}^4 \mathbb{E}_{\pi_{TI}^{(4)}} Q_i(t) \\ < \infty, \end{aligned} \tag{29}$$

■

As mentioned before, $\tilde{\pi}_2^{(4)}$ is defined by simply replacing checking the status of Queue 3 with that of Queue 2 and vice versa. A similar proof of throughput optimality holds for this policy as well. In fact, it is easily seen that when N is even, every policy $\pi \in \tilde{\Pi}^{(N)}$ has a dual that breaks ties by prioritizing an alternate set of queues than π . With $N = 4$, we obtained two policies that prioritized Queue 2 and Queue 3 respectively.

Remark. In the above proof, the manner in which $\pi_{TD}^{(3)}$ and $\pi_{IQ}^{(3)}$ were spliced to form $\pi_{TI}^{(4)}$ is of particular note. While $\pi_{TD}^{(3)}$ is a policy that prioritizes “outer” queues (Queue 1), $\pi_{IQ}^{(3)}$ prioritizes “inner” queues, i.e., Queue 2. Since, in the four queue system, Queue 2 becomes the outer and inner queue for these two policies respectively, both subsystems are simultaneously stabilized.

2) *Analysis of $\tilde{\pi}_3^{(4)}$ and $\tilde{\pi}_4^{(4)}$:* Table. II compares the activation vectors of policies $\pi_{TI}^{(4)}$, $\tilde{\pi}_1^{(4)}$ and $\tilde{\pi}_3^{(4)}$. Note that in column 4, only the places where $\tilde{\pi}_3^{(4)}$ differs from $\tilde{\pi}_1^{(4)}$ are shown in blue. The unspecified entries, therefore, follow those in the corresponding rows of the third column. Also note that the first coordinate of the first column is $i_4(t)$ and not $i_1(t)$.

TABLE II: Comparison of $S(t)$ under $\pi_{TI}^{(4)}$, $\tilde{\pi}_1^{(4)}$ and $\tilde{\pi}_3^{(4)}$

$[i_4(t), i_3(t), i_2(t), i_1(t)]$	$\pi_{TI}^{(4)}$	$\tilde{\pi}_1^{(4)}$	$\tilde{\pi}_3^{(4)}$
0000	1001	1001	
0001	1001	1001	
0010	1010	1010	
0011	1010	1010	
0100	0101	0101	
0101	0101	0101	
0110	1010	1010	1010
0111	1010	0101	0101
1000	1001	1001	
1001	1001	1001	
1010	1010	1010	
1011	1010	1010	
1100	0101	0101	
1101	0101	0101	
1110	1010	1010	
1111	1010	1010	0101

Proposition 18. $\tilde{\pi}_3^{(4)}$ and $\tilde{\pi}_4^{(4)}$ are Throughput Optimal.

PROOF. The proof essentially proceeds by using a modification of the technique used to prove the delay optimality of $\tilde{\pi}_{IQ}^{(3)}$. This is eventually used to show that the total system backlog with $\tilde{\pi}_3^{(4)}$ and $\tilde{\pi}_1^{(4)}$ are identically distributed, and since $\tilde{\pi}_1^{(4)}$ is throughput-optimal, so is $\tilde{\pi}_3^{(4)}$. The proof is available in Sec. XIV-M in the Appendix. ■

B. Analysis of Delay in $\Pi^{(4)}$

We now show that $\Pi^{(4)}$ **does not contain** any queue length agnostic policy that is *uniformly* delay optimal over the entire set Λ_4^o . This is unlike the case with $N = 3$, where $\tilde{\pi}_{IQ}^{(3)}$ produced the lowest possible delay regardless of the arrival rate. We first prove in Prop. 19 that $\tilde{\Pi}^{(4)}$ does not contain any uniformly delay optimal policy.

Proposition 19. There exist arrival rate vectors λ_1 and λ_2 within Λ_4^0 , such that, under λ_1 ,

$$\begin{aligned} \sum_{i=1}^4 Q_i^{\tilde{\pi}_1^{(4)}}(t) &\leq \sum_{i=1}^4 Q_i^{\tilde{\pi}_2^{(4)}}(t), \quad \forall t \geq 0, \text{ and,} \\ \sum_{i=1}^4 Q_i^{\tilde{\pi}_3^{(4)}}(t) &\leq \sum_{i=1}^4 Q_i^{\tilde{\pi}_4^{(4)}}(t), \quad \forall t \geq 0, \end{aligned} \quad (30)$$

and under λ_2 ,

$$\begin{aligned} \sum_{i=1}^4 Q_i^{\tilde{\pi}_2^{(4)}}(t) &\leq \sum_{i=1}^4 Q_i^{\tilde{\pi}_1^{(4)}}(t), \quad \forall t \geq 0, \text{ and,} \\ \sum_{i=1}^4 Q_i^{\tilde{\pi}_4^{(4)}}(t) &\leq \sum_{i=1}^4 Q_i^{\tilde{\pi}_3^{(4)}}(t), \quad \forall t \geq 0. \end{aligned} \quad (31)$$

PROOF. The proof can be found in Sec. XIV-N of the Appendix. ■

Next, we show that policies in $\tilde{\Pi}^{(4)}$ show better delay performance than those in $\Pi_M^{(4)}$.

Proposition 20. Given any policy $\pi \in \Pi_M^{(4)} \setminus \tilde{\Pi}^{(4)}$, there exists a policy $\pi' \in \tilde{\Pi}^{(4)}$ such that

$$\sum_{i=1}^4 Q_i^{\pi'}(t) \leq \sum_{i=1}^4 Q_i^{\pi}(t), \quad \forall t \geq 0. \quad (32)$$

PROOF. Proof available in Sec. XIV-O of the Appendix. ■

We already know, from Eqn.(17), that the delay of any policy in $\Pi^{(N)}$ can be improved by projecting it onto $\Pi_M^{(N)}$. Now, Prop. 20, along with Eqn.(17), 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 $\tilde{\Pi}^{(4)}$ (Prop. 19), has one far reaching consequence.

Theorem 21. For all $N \geq 4$, there does not exist any policy in $\Pi_M^{(N)}$ that is uniformly delay optimal over all of Λ_N^o .

PROOF. Suppose $\pi \in \Pi_M^{(N)}$ was a delay optimal policy for the N queue system for some N , then, by setting $\lambda_i = 0$, for all $i \geq 5$ we would get a delay optimal policy for the 4 queue system, contradicting Prop. 19. Hence, such a policy cannot exist. ■

Remark. A few important remarks are in order.

- 1) It is possible that policies which take decisions based on $\mathbf{Q}(t)$ rather than just $\zeta(t)$ might perform better. MaxWeight (MW) and its projection into $\Gamma_M^{(N)}$, viz $L(MW)$, use all of $\mathbf{Q}(t)$ and hence, perform dynamic randomization between s_1 and s_2 when $\zeta(t) = [0, 1, 1, 0]$, based on which queue is larger. Their performance with respect to the policies developed in this section is further explored through simulations in Sec. XII.
- 2) It is to be noted that, as Column 3 in Table. V shows, $\{\tilde{\pi}_i^{(4)}, 1 \leq i \leq 4\}$ satisfy Condition 3 in Lem. 1. It appears that prioritizing “inner,” and hence more constrained, queues improves not only throughput but also delay.

So in essence, Thm. 21 shows that while *throughput* optimality in these interference graphs only requires knowledge of queue occupancy (i.e., $\zeta(t)$), *delay* optimality potentially requires more information from the filtration, $\mathcal{H}_t := \sigma(\{\mathbf{Q}(t), \dots, \mathbf{Q}(0), \mathbf{s}(t-1) \dots \mathbf{s}(0)\})$. We explore this in greater detail in Sec. XII where we compare the performance of these policies with those of MW and $L(MW)$.

X. CLUSTER-OF-CLIQUE 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 the Star-of-Cliques Network

Consider the following policy that we denote $\tilde{\phi}_{IC}^{(S)}$, which is motivated by the 3-node path graph policy $\tilde{\pi}_{IO}^{(3)}$ which we discussed in Sec. V-B. 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, $\tilde{\phi}_{IC}^{(S)}$ maps the occupancy vector $\zeta(t) \in \{0, 1\}^N$ to an activation vector $\mathbf{s}(t) \in \{0, 1\}^N$. We define $\tilde{\phi}_{IC}^{(S)}$ as follows.

At each time t :

- 1) If $\prod_{m=2}^N (\sum_{l \in C_m} \zeta_l(t)) > 0$ 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 if, at time t ,

- 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 22. $\tilde{\phi}_{IC}^{(S)}$ 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. 2 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 XIV-P for details. ■

Towards the end of our discussion on queue nonemptiness-based scheduling for path-graph networks with $N = 3$ queues (see Sec. V-E), we defined a *non-MSM* policy $\pi_{IQ}^{(3)}$. Extending this to the SoC network model gives us a second queue nonemptiness-based policy $\phi_{IC}^{(S)}$, 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 23. $\phi_{IC}^{(S)}$ is throughput-optimal.

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

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. XI 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 24. 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,

$$\sum_{m=1}^N \sum_{j=1}^{N_m} Q_{m,j}^{\tilde{\phi}_{IC}^{(S)}}(t) \leq_{st} \sum_{m=1}^N \sum_{j=1}^{N_m} Q_{m,j}^{\phi_{IC}^{(S)}}(t), \quad \forall t \geq 0. \quad (33)$$

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

We now begin our study of scheduling in LAoC networks. However, as mentioned before, we will return to these policies 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. VII. 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. 5a 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 25. $\theta_{TD}^{(3L)}$ 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. XIV-S in the Appendix. ■

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

PROOF. The policy $\theta_{SP}^{(5L)}$ is formed by splicing together $\theta_{TD}^{(3L)}$ and $\theta_{BU}^{(3L)}$. This means that $\theta_{SP}^{(5L)}$ restricted to Queues 1 to 3 is $\theta_{BU}^{(3L)}$ and restricted to Queues 3 to 5 is $\theta_{TD}^{(3L)}$. The throughput optimality of $\theta_{TD}^{(3L)}$ and $\theta_{BU}^{(3L)}$ means that for every $\lambda \in \Lambda_l^{(5)}$ (defined in (12)),

$$\limsup_{T \rightarrow \infty} \frac{1}{T} \sum_{t=0}^{T-1} \sum_{i=3}^5 \sum_{j=1}^{N_i} \mathbb{E}_{\theta_{TD}^{(3L)}} Q_{i,j}(t) < \infty, \text{ and} \quad (34)$$

$$\limsup_{T \rightarrow \infty} \frac{1}{T} \sum_{t=0}^{T-1} \sum_{i=1}^3 \sum_{j=1}^{N_i} \mathbb{E}_{\theta_{BU}^{(3L)}} Q_{i,j}(t) < \infty. \quad (35)$$

Now, as in Thm. 12, for every $\lambda \in \Lambda_l^{(5)}$ we define two new arrival rate vectors $\lambda_{1:3} = [\lambda_1, \lambda_2, \lambda_3]$ and $\lambda_{3:5} = [\lambda_3, \lambda_4, \lambda_5]$ and note that by the definition of the set $\Lambda_l^{(N)}$, $\lambda_{1:3}$ and $\lambda_{3:5}$ are both in $\Lambda_l^{(3)}$, which means that (34) and (35) are still separately true. We conclude the proof by observing that

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

■

To summarize, in this section, we studied scheduling in the Star-of-Cliques and Linear-Array-of-Cliques models that occur frequently in IoT-type sensor network applications. Having characterized the capacity region of such networks, we proposed an analysed multiple scheduling policies. However, as mentioned before, these policies depend on being able to find a nonempty

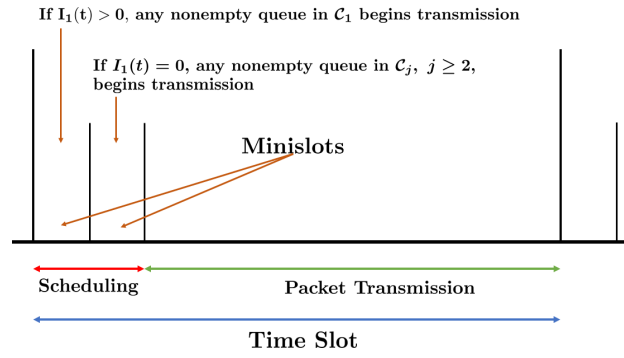


Fig. 11: Implementing $\phi_{IC}^{(S)}$ using the slot and minislot structures. It is important to note that the number of minislots is $O(1)$, i.e., *does not scale* with the number of communication links the network. This results in constant scheduling overhead, which is patently desirable.

queue in every slot in which the system is not empty. While disseminating this occupancy information across the network is certainly not as expensive as sharing queue length information (required by the MaxWeight family of scheduling algorithms) it would be beneficial to sensor network designers if we could produce scheduling policies that worked with even less information. In the following section, we attempt to do precisely that.

C. Preamble to the rest of the Paper

Since we will be introducing several scheduling policies in the sequel, we will now present a short preamble to the rest of the paper for the convenience of the reader. Note that all the necessary notation is also present in the glossary of notation XIV-B that the reader can refer to for clarification.

- We first propose queue nonemptiness-based policies for SoC and LAoC networks. We begin with SoC networks and propose two policies $\phi_{IC}^{(S)}$ and $\tilde{\phi}_{IC}^{(S)}$ that are based on the policies $\pi_{IQ}^{(3)}$ and $\tilde{\pi}_{IQ}^{(3)}$ respectively, that were proposed and analysed in Sections V-B and V-E. 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 . We analyse their throughput and delay properties.
- We then move on to LAoC networks for which we, once again, propose three policies. We begin with 3-clique networks and propose extensions of the Top-Down and Bottom-Up policies we developed in Sec V-A. We call these policies $\theta_{TD}^{(3L)}$ and $\theta_{BU}^{(3L)}$, where the “L” in the superscript stands for LAoC network.
- In the sequel, ϕ will always represent a policy for SoC networks and θ for LAoC network (recall that π was used to denote policies on path-graph networks).
- We then move on to decentralized implementation, introduce the concept of “minislots” show how they can be used to implement $\phi_{IC}^{(S)}$ in a distributed fashion. We then propose a class of policies that we denote $\phi_{IC}^{(S)}(T)$ for SoC networks, that require knowledge of the occupancy vector $\zeta = [\mathbb{I}_{\{Q_1(t)>0\}}, \dots, \mathbb{I}_{\{Q_N(t)>0\}}]$ only every T time slots and show that the class is throughput optimal. Following this, we propose a policy $\phi_{CS}^{(S)}$ that¹⁰, like the QZMAC protocol in [20], takes scheduling decisions based solely on the information gathered by sensing the channel for activity.

XI. SOME REMARKS ON DECENTRALIZED IMPLEMENTATION

In this section, we discuss several ways in which the policies developed and analysed hitherto can be made amenable to decentralized implementation. To accomplish our stated objective of constructing a distributed scheduling protocol, we take the help of what are known as *minislots* [20], [32] which we describe in detail, below.

Transmission Sensing: We assume that all nodes transmit at the same fixed power, and 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 clique C_1 and vice versa, as dictated by the interference constraints¹¹. 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 interval (akin to the “Clear Channel Assessment” or CCA mechanism [33]). 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 [20] and Fig. 11).

Decentralized methods of implementing both $\phi_{IC}^{(S)}$ and $\tilde{\phi}_{IC}^{(S)}$ immediately follow from the minislot structure. 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 first discuss implementing $\phi_{IC}^{(S)}$ in detail, and then $\tilde{\phi}_{IC}^{(S)}$ in Sec. XI-C.

¹⁰The “CS” in the subscript stands for channel sensing.

¹¹Obviously, for all $2 \leq j, k \leq N$, nodes in C_j cannot sense transmissions in C_k and vice versa.

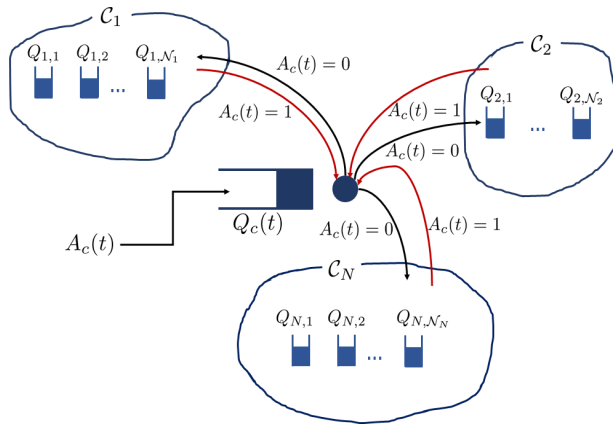


Fig. 12: The system upon which the policy $\phi_{CS}^{(S)}$ is analysed. $Q_c(t)$ is the central queue which has highest priority. This means that the queue is served whenever it has packets, i.e., whenever the arrival $A_c(t) = 1$. This is indicated by the **red arrows** that show access to the channel 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**.

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. 11). 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 in later in this section. In Sec. XI-B we discuss a version of $\phi_{IC}^{(S)}$ that requires no explicit information exchange between queues.

1) $\phi_{IC}^{(S)}$ with Periodic Occupancy Information: The $\phi_{IC}^{(S)}(T)$ Policy: Suppose the nodes share their occupancy status at slots $\{kT, T \geq 1, k \in \mathbb{N} \cup \{0\}\}$. By setting T large enough, the cost of exchanging this information can be amortized over this interval. We group slots $kT, kT + 1, \dots, kT + T - 1$ into the k^{th} frame ($k \geq 0$). In each frame k , only the packets that have arrived *until* time kT are served and any new packets arriving in the frame are queued. With this, we get a new *class* of protocols parameterized by the information sharing interval T , denoted $\{\phi_{IC}^{(S)}(T), T \geq 1\}$, defined as follows

Over $kT, \dots, kT + T - 1$,

- 1) If $I_1(kT) > 0$, serve nonempty queues in C_1 (in any arbitrary order) until either the next frame begins or all the packets in C_1 have left the system. The other cliques detect this when they sense no power in the first minislot of such a slot.
- 2) Any slots left after C_1 is served, is used by the other cliques for transmission until either frame $k + 1$ begins or they run out of packets.

We prove that $\phi_{IC}^{(S)}(T)$ is **throughput optimal** in Sec. XIV-Q in the Appendix and therein also show some desirable delay properties of the protocol. While $\phi_{IC}^{(S)}(T)$ requires state information dissemination only every T slots, we would like to explore the possibility of implementing $\phi_{IC}^{(S)}$ without *any* explicit information dissemination.

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 packet¹². 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 transmitted¹³ and its backlog $Q_i(t - V_i(t))$ at that instant. With this partial information structure, we have already shown, in [20], 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

¹²The backlog information could be quantized and contained in the packet header, for example.

¹³If 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.

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 information version $\phi_{CS}^{(S)}$, of $\phi_{IC}^{(S)}$ 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. 12).

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. 11) 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 27. The policy $\phi_{CS}^{(S)}$ is throughput-optimal.

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

We now move on to $\tilde{\phi}_{CS}^{(S)}$, the low-delay counterpart of $\phi_{CS}^{(S)}$. We now show how to use the minislot structure to implement $\tilde{\phi}_{CS}^{(S)}$ in a decentralized manner.

C. Decentralized Implementation of $\tilde{\phi}_{IC}^{(S)}$

While $\phi_{CS}^{(S)}$ works even without a sink node in each clique to determine which node transmits in its clique in each slot, this implementation of $\tilde{\phi}_{IC}^{(S)}$ requires a sink. This protocol, while throughput-optimal, is tailored for **heavy-traffic** applications, when the queues in the system are rarely empty. The protocol requires three minislots and proceeds as follows.

At time t

- 1) Every empty queue in clique C_j , $j \geq 2$, makes a small transmission during minislot 1. The sink uses this to determine if its clique is empty, i.e., if $I_j(t) = 0$.
- 2) In minislot 2, every peripheral sink with $I_j(t) = 0$, $j \geq 2$ makes a small transmission which is sensed by the sink of Clique 1.
- 3) We now have three possibilities:
 - Suppose one of the outer cliques is empty and $I_1(t) > 0$. Then C_1 senses power in minislot 2 infers that at least one outer clique is empty and begins transmission in minislot 3.
 - If none of the outer cliques is empty then the outer cliques do not sense any power in minislot 3 and those cliques that have nonempty queues begin transmission.
 - Finally, if one of the outer cliques is empty but $I_1(t) = 0$ then, once again, the outer cliques do not sense any power in minislot 3 and begin transmission.

Remark. Prop. 24 shows that policy $\tilde{\phi}_{IC}^{(S)}$ induces stochastically smaller delays than $\phi_{IC}^{(S)}$. However, implementing the former consumes more energy than implementing $\phi_{IC}^{(S)}$ and its partial knowledge versions, since even empty nodes need to be ON. This is why the present protocol is more suited to heavy-traffic applications, when the arrival rates to the system are heavy enough that queues do not remain empty for long.

D. Policies with Periodic State Information for Linear-Arrays-of-Cliques

We now move on to proposing a protocol similar to $\phi_{IC}^{(S)}(T)$ which we discussed in Sec. XI-A1, that does not require the empty-nonempty status of queues in the system in every time slot, for LAoC networks. Once again, we assume the presence of a sink node in each clique, that periodically aggregates and disseminates this information from the nodes in its own clique to take scheduling decisions. Suppose the system receives the occupancy information of cliques only every T time slots, i.e., information about whether there exists a non empty queue in clique i ($1 \leq i \leq 3$) is disseminated only at kT , $k \in \{0, 1, 2, \dots\}$. Then we would like to know if $\theta_{TD}^{(3L)}$ and $\theta_{BU}^{(3L)}$ (both defined in Sec. X-B) can be suitably modified to ensure stability. Thereafter, our splicing technique will immediately provide us with policies for systems with $N = 4$ and $N = 5$ cliques.

Towards that end, first define for every slot t the *clique occupancy vector* $\zeta(t) := [i_1(t), i_2(t), i_3(t)]$, where $\zeta_j(t) := \mathbb{I}_{\{\sum_{k=1}^{N_j} i_{j,k}(t) > 0\}}$ indicates whether Clique j is empty or not. We call the slots $kT, kT + 1, \dots, kT + (T - 1)$ together, the k^{th} frame. Similar to $\phi_{IC}^{(S)}(T)$ consider the class of policies $\theta_{TD}^{(3L)}(T)$ indexed by the frequency of information dissemination, T , and defined as follows. Note that these policies only require knowledge of $\zeta(kT)$, $k = 0, 1, \dots$.

During Frame k , i.e., over slots $\{t : kT \leq t \leq kT + (T - 1)\}$

- If $\zeta_1(kT) = 1$ serve queues in C_1 until either T slots elapse or all queues in C_1 become empty, whichever occurs first.
 - Simultaneously serve any non empty queues in C_3 .
- Else, serve queues in C_2 ,
 - if $\zeta_1(kT) = 1$ but C_1 became empty at some $t < kT + (T - 1)$ and $\zeta_2(kT) = 1$, or
 - if $\zeta_1(kT) = 0$ and $\zeta_2(kT) = 1$.

In both of the above cases, C_2 is served either the frame ends or all queues in C_2 become empty, whichever occurs first.

- Else, serve non empty queues in C_3 (if any).

$\theta_{BU}^{(3L)}(T)$ is defined as above, replacing C_3 with C_1 and vice versa.

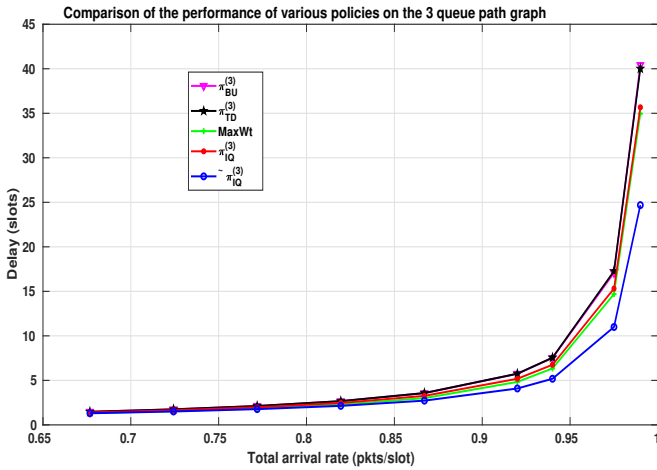
Remark. $\theta_{TD}^{(3L)}(T)$ and $\theta_{BU}^{(3L)}(T)$ are both throughput-optimal.

PROOF. The proof is available in Sec. XIV-V in the Appendix. ■

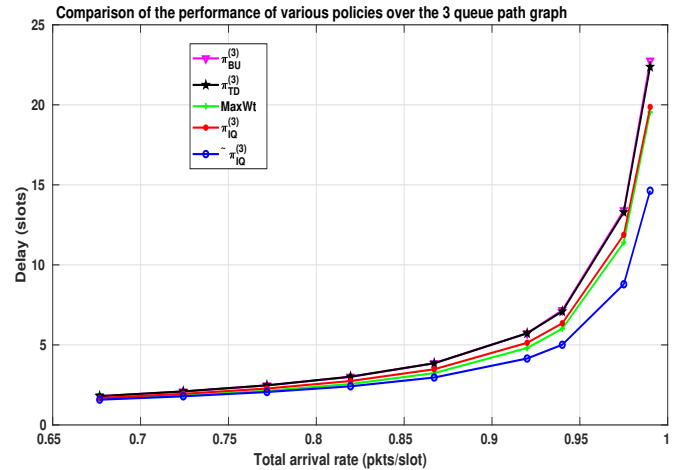
Clearly, splicing these two policies yields a periodic state information-based policy for 5-clique and hence, 4-clique, LAoCs. This policy is obviously throughput optimal.

XII. SIMULATION RESULTS

In this section we numerically compare the performance of the various policies we have proposed and analysed 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. V and compare them against the MaxWeight scheduling policy. To recapitulate, $\pi_{TD}^{(3)}$ and $\pi_{BU}^{(3)}$ are the Top-Down and Bottom-Up policies respectively, $\tilde{\pi}_{IQ}^{(3)}$ is the delay optimal policy defined in V-B and $\pi_{IQ}^{(3)}$ is the throughput-optimal *non-MSM* policy defined in V-E. 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 13a and 13b. As claimed in Thm. 5, $\tilde{\pi}_{IQ}^{(3)}$ 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 $\pi_{TD}^{(3)}$. Notice that in both plots MaxWeight does not perform as well as $\tilde{\pi}_{IQ}^{(3)}$ showing that it does not prioritize the middle queue frequently “enough.”



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



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

Fig. 13: 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. V are shown in Figures (a) and (b), and compared with the MaxWeight scheduling policy [10].

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. 5, i.e., the delay optimality of policy $\tilde{\pi}_{IQ}^{(3)}$, 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 $A_i(t) = 0$

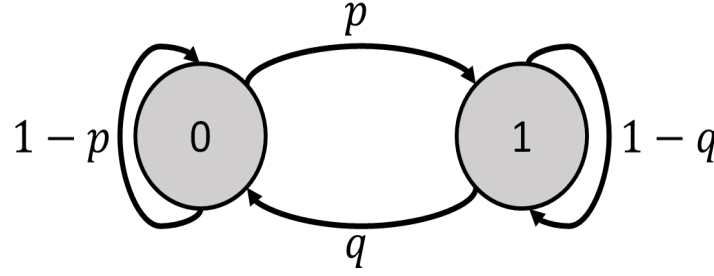
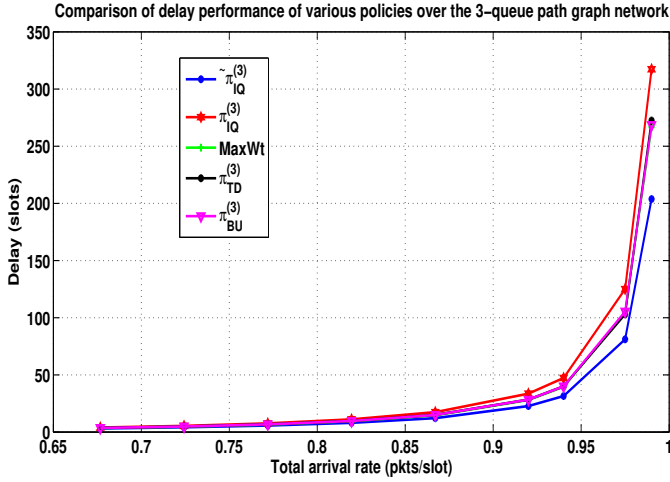
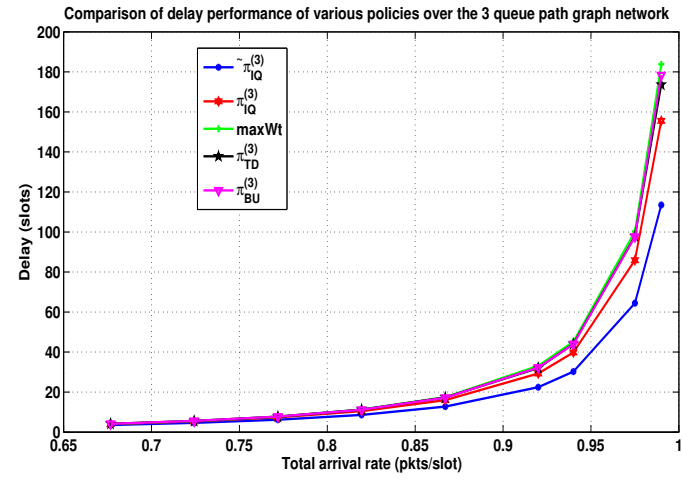


Fig. 14: The transition probability diagram of the Markovian arrival process. If, in slot t , the arrival process was in State 0, i.e., $A(t) = 0$, then in slot $t + 1$, $A(t + 1) = 1$ with probability p and $A(t + 1) = 0$ with probability $1 - p$.

refers to no arrivals. Fig. 14 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 $t \geq 1$, $\mathbb{E}A_j(t) = \xi_{1,j} = \lambda_j$.



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



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

Fig. 15: 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. 14) are chosen as follows $p_i = 0.10$, and $q_i = (\frac{1}{\lambda_i} - 1)p_i$.

Both plots (Figures 15a and 15b) bear out the fact that even with non-Bernoulli arrivals, $\tilde{\pi}_{IQ}^{(3)}$ 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 $\rho_\gamma^{(3)}$, indexed by the randomization parameter $\gamma \in [0, 1]$. In Sec. VI we derived an inner bound on the set of arrival rates that the policy can stabilize for a given γ , its stability region¹⁴, and showed that $\Lambda_\gamma^{(3)} \nearrow \Lambda_3$ as $\gamma \uparrow 1$. The plot in Fig. 16, simulated with $\gamma = 0.5, 0.55$, and 0.6 help corroborate our analysis. However, the plots also suggest that the inner bound $\Lambda_\gamma^{(3)}$ 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. VII-0a 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 III, where we compare our proposed policies 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. V-B) to project a modification of MaxWeight (*MW*) called $MW\alpha$ onto $\Gamma_M^{(N)}$. The $MW\alpha$ policy, studied in [34] and [21], 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 [35].

The table shows that our proposed policies do outperform MaxWeight in all cases¹⁵. Note that the arrival rate vectors have

¹⁴See Sec. II-A for details.

¹⁵The construction of the QNB-MSM policy in the first row which, by our nomenclature, is denoted $\tilde{\pi}^{(5)}$, is discussed in depth in [36] and serves as a good example to illustrate the general splicing process discussed in VII-0a.

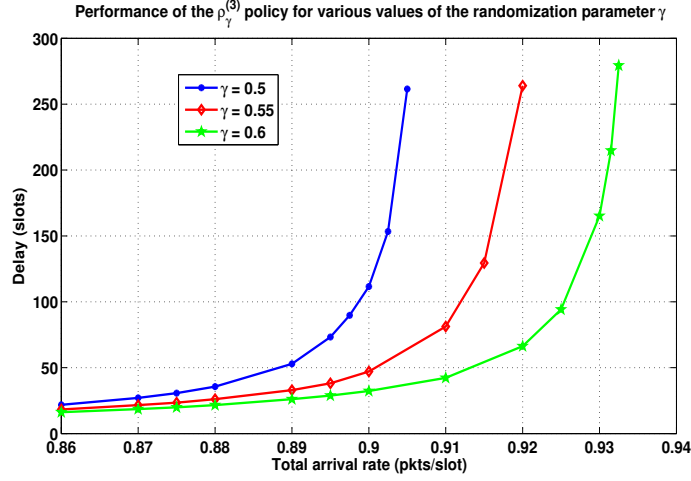


Fig. 16: Illustrating the loss of stability due to the “Flow-in-the-Middle” problem discussed in Sec. VI. We compare the delay performance of the policy $\rho_\gamma^{(3)}$ along the trajectory $\lambda(s) = s \times [0.74, 0.25, 0.74]$, $s \in [0, 1]$, in the capacity region Λ_3 . For every $s \leq 1$, 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.

Number of queues (N)	Mean sum queue length (packets) Bernoulli arrivals		
	QNB-MSM ($\tilde{\pi}^{(N)}$)	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 III: 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. XIV-W of the Appendix.

not been shown in Table III due to space constraints. We have reported the vectors in Sec. XIV-W 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. VII-0a) 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. 21, 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. 17, 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. 5a. It consists of 4 cliques and a total of 9 queues. Table IV shows the result of simulating $\tilde{\phi}$, $\tilde{\theta}^{(SL)}$ (the projected version of $\theta_{SP}^{(SL)}$, defined in Sec. X-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.

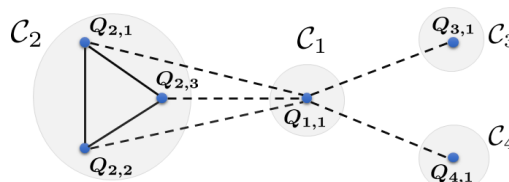


Fig. 17: The Star-of-Cliques (SoC) network used to study the performance of $\tilde{\phi}$. Simulation results are reported in Table IV.

Cluster of Cliques Network	Mean sum queue length (packets)	
	QNB-MSM	MaxWeight
Star of Cliques (Fig. 17)	45.535	57.861
Linear Array of Cliques (Fig. 5a)	245.038	309.450

TABLE IV: Comparison of sum queue length under the proposed Cluster of Cliques policies, and MaxWeight acting on the networks in Fig. 5a and Fig. 17. Details about the arrival rate vectors can be found in Sec. XIV-W of the Appendix.

XIII. CONCLUSION AND FUTURE WORK

In this paper, we began by studying the scheduling of transmissions over a class of non collocated 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 MSM-QNB policies on path-graphs with 3 queues and showed that it 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 and studied their stability and delay properties. We showed how the minislot structure can be used to implement these policies in a decentralized manner, and also developed a protocol that requires no explicit exchange of even occupancy information and proved that it is throughput-optimal. Our simulation results showed that the QNB policies we have developed, in fact, 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, obtaining better bounds on the stability region of the policy $\rho_y^{(3)}$ and proving the throughput-optimality of the Top-Down and Bottom-Up policies for general N -queue path graph networks. We would like to find techniques to encode occupancy information $\zeta(t)$ using fewer than N bits, getting as close as possible to $\log_2(N)$ bits, albeit in a decentralized manner. Finally, we would also like 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.

XIV. APPENDIX

A. Glossary of Acronyms

- BU: Bottom-up
- CoC: Cluster-of-Cliques
- CSMA: Carrier Sense Multiple Access
- D.O.: Uniformly Delay Optimal (see Defn. V-B).
- 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 [10].
- $MW\alpha$: The MaxWeight- α algorithm defined in [34]
- 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) $\zeta(t)$: the occupancy vector or nonemptiness vector at time t , defined as $\zeta(t) = [\mathbb{I}_{\{Q_1(t)>0\}}, \dots, \mathbb{I}_{\{Q_N(t)>0\}}]$.
 - 3) \mathcal{V} : is the set of all activation vectors. Clearly, in a system of N queues, $\mathcal{V} \subseteq \{0, 1\}^N$ 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 $\zeta(t)$ 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) $|A|$: represents the cardinality of set A .
 - 11) $\text{Geo}(\lambda)/\text{Geo}(\mu)/1$ 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) $\{\tilde{\pi}_i^{(4)}, 1 \leq i \leq 4\}$: 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. X-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. X-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. X, 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 ζ only every T time slots.
 - 21) $\phi_{CS}^{(S)}$: that, like the QZMAC protocol in [20], 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. 2, i.e., throughput optimality of policies satisfying property \mathcal{P} , proceeds via a Lyapunov argument. The sole purpose of 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 $\lambda \in \mathbb{R}_+^N$ 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

$$L(\mathbf{Q}(t)) := \left(\sum_{i=1}^N Q_i(t) \right)^2 = Q^2(t). \quad (36)$$

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

$$Q^2(t+1) \leq Q^2(t) + D^2(t) + A^2(t) - 2Q(t)(D(t) - A(t)).$$

Hence, the expected single slot drift becomes

$$\begin{aligned}
& \mathbb{E} \left[L(\mathbf{Q}(t+1)) - L(\mathbf{Q}(t)) \mid \mathbf{Q}(t) = \mathbf{q} \right] \\
& \stackrel{\dagger 1}{\leq} \mathbb{E} \left[D^2(t) + A^2(t) \mid \mathbf{q} \right] - 2q \mathbb{E} \left[D(t) - A(t) \mid \mathbf{q} \right] \\
& \stackrel{\dagger 2}{=} 1 + N^2 - 2q \left(\mathbb{E} \left[D(t) \mid \mathbf{q} \right] - \sum_{i=1}^N \lambda_i \right),
\end{aligned} \tag{37}$$

where in $\dagger 1$, $\mathbf{q} = [q_1, \dots, q_N]$, and $q = \sum_{i=1}^N q_i$. In $\dagger 2$ 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, $\mathbb{E} \left[A(t) \mid \mathbf{q} \right] = \mathbb{E} A(t) = \sum_{i=1}^N \lambda_i$. Since the policy schedules a non empty queue in every slot,

$$\begin{aligned}
\mathbb{E} \left[D(t) \mid \mathbf{q} \right] &= \mathbb{I}_{\{Q(t) > 0\}}, \text{ i.e.,} \\
\mathbb{E} \left[\sum_{i=1}^N D_i(t) \mid \mathbf{q} \right] &= \mathbb{I}_{\{\sum_{i=1}^N Q_i(t) > 0\}}
\end{aligned} \tag{38}$$

Taking expectations on both sides of (37), we see that

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

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

$$\mathbb{E} Q^2(t+1) - \mathbb{E} Q^2(t) \leq 1 + N^2 - 2\epsilon \mathbb{E} Q(t). \tag{39}$$

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

$$\mathbb{E} Q^2(T) - \mathbb{E} Q^2(0) \leq T(1 + N^2) - 2\epsilon \sum_{t=0}^{T-1} \mathbb{E} Q(t). \tag{40}$$

A little bit of algebra shows that

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

which implies strong stability. In the sequel, we will call this rather standard technique [22] of showing strong stability, the *telescoping sum method*. As a precursor to the proof of Lem. 2, observe that this class of policies satisfies property \mathcal{P} in the lemma, i.e., (\mathcal{P}) since $\mathbb{E} \left[D(t) \mid \mathbf{q} \right] = \mathbb{I}_{\{Q(t) > 0\}}$ means that $\sum_{i=1}^N D_i(t) = 0 \iff \sum_{i=1}^N Q_i(t) = 0$. \square

D. Proof of Thm. 3

Queues 1 and 2 form a priority queue and $\pi_{TD}^{(3)}$ serves the pair of queues 1 and 2 whenever either of them is nonempty. So, $\pi_{TD}^{(3)}$ satisfies Property \mathcal{P} for $i = 1$ (specifically, Eqn. (\mathcal{P})), which means that the process $\{[Q_1(t), Q_2(t)], t \geq 0\}$, 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 $Q_1(0) \leq 1$, in any time slot, Queue 1 has at most 1 packet, which is the arrival during that slot, i.e., $Q_i(t) = A_i(t)$, $\forall t \geq 1$. 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 XIV-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, $\{[Q_1(t), Q_2(t)], t \geq 0\}$ forms an aperiodic, irreducible positive recurrent DTMC. This means that there exists a steady state probability measure on Queue 2's backlog such that,

$$\begin{aligned} \lim_{t \rightarrow \infty} P\{Q_2(t) = 0\} &= 1 - \lambda_2 \mathbb{E}B_2 \\ &= 1 - \frac{\lambda_2}{1 - \lambda_1}. \end{aligned} \quad (41)$$

From the definition of $\pi_{TD}^{(3)}$ 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,

$$\begin{aligned} P\{S_3(t) = 1\} &= P\{Q_1(t) > 0\} + P\{Q_1(t) = 0, Q_2(t) = 0\} \\ &= \lambda_1 + P\{Q_2(t) = 0\}P\{Q_1(t) = 0 \mid Q_2(t) = 0\} \\ &\stackrel{*1}{=} \lambda_1 + \left(1 - \frac{\lambda_2}{1 - \lambda_1}\right)P\{A_1(t) = 0 \mid Q_2(t) = 0\} \\ &\stackrel{*2}{=} \lambda_1 + \left(1 - \frac{\lambda_2}{1 - \lambda_1}\right)(1 - \lambda_1) \\ &= 1 - \lambda_2 > \lambda_3. \end{aligned} \quad (42)$$

Equality *1 uses Eqn. (41) 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 $L : \mathbb{N} \rightarrow \mathbb{R}_+$ as $L(Q_3(t)) = Q_3^2(t)$.

$$\begin{aligned} \mathbb{E}[L(t+1) - L(t)] &= \mathbb{E}[Q_3^2(t+1) - Q_3^2(t)] \\ &= \mathbb{E}[(Q_3(t) - S_3(t))^+ + A_3(t+1)]^2 - Q_3^2(t) \\ &\stackrel{\star 3}{\leq} 2 - 2\mathbb{E}Q_3(t)\mathbb{E}[S_3(t) - A_3(t+1)] \\ &= 2 - 2\mathbb{E}Q_3(t)(1 - \lambda_2 - \lambda_3) \\ &\stackrel{\star 4}{=} 2 - 2\delta\mathbb{E}Q_3(t) \end{aligned} \quad (43)$$

In $\star 3$, we have once again used the fact that for any four non negative reals w, x, y and z , with $w \leq (x - y)^+ + z$, $w^2 \leq x^2 + y^2 + z^2 - 2x(y - z)$, with $w = Q_3(t+1)$, $x = Q_3(t)$, $y = S_3(t)$ and $z = A_3(t+1)$. Further, $S_3(t) \leq 1$ and $A_3(t+1) \leq 1$. 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

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

the system is also strongly stable under this policy. The other policy $\pi_{BU}^{(3)}$ simply swaps the priorities of Queues 1 and 3 in the enumeration V-A, and its proof proceeds as before, mutatis mutandis. \square

E. Proof of Thm. 4

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, one 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 $\tilde{\pi}_{IQ}^{(3)}$ satisfies property \mathcal{P} , and from Lem. 2, $\tilde{\pi}_{IQ}^{(3)}$ is T.O. \square

F. Proof of Thm. 5

We adapt the technique used in Lem. 4.2 of [15] to prove this result¹⁶. The main idea is to construct a sequence $\{\pi'_k, k \geq 0\}$ of intermediate policies such that the backlog in every queue converges sample path-wise to that of $\tilde{\pi}_{IQ}^{(3)}$ which means that for every $t \geq 0$,

$$\lim_{k \rightarrow \infty} Q_l^{\pi'_k}(t) = Q_l^{\tilde{\pi}_{IQ}^{(3)}}(t), \quad 1 \leq l \leq 3, \quad (44)$$

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 $\tilde{\pi}_{IQ}^{(3)}$, $\{\pi'_k, k \geq 0\}$ and π act (by assumption, the initial conditions are equal, i.e., on every sample path $\mathbf{Q}^\pi(0) = \mathbf{Q}^{\tilde{\pi}_{IQ}^{(3)}}(0) = \mathbf{Q}^{\pi'_k}(0) = \mathbf{Q}$, $\forall k \geq 0$, 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 $\tilde{\pi}_{IQ}^{(3)}$ which means the activation vectors chosen by the two policies are the same. In other words, $\mathbf{s}^{\pi'_0}(0) = \mathbf{s}^{\tilde{\pi}_{IQ}^{(3)}}(0)$, 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. 1.

- 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, $Q_l^{\pi'_0}(1) = Q_l^\pi(1) + 1 \Rightarrow Q_l^{\pi'_0}(0) = Q_l^\pi(0) = Q_l \neq 0$, 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, $\tilde{\pi}_{IQ}^{(3)}$ 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 $Q_2^{\pi'_0}(0) = Q_2^\pi(0) - 1$.
- $l = 2 \Rightarrow Q_1(0) \neq 0$ and $Q_3(0) \neq 0$, since by definition, $\tilde{\pi}_{IQ}^{(3)}$ 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 $Q_1^{\pi'_0}(1) = Q_1^\pi(1) - 1$ 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, $\tilde{\pi}_{IQ}^{(3)}$ 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 [15] and will not be repeated. For every $k \geq 0$, π'_{k+1} is defined as the policy that follows π'_{k+1} over slots $0, 1, \dots, k$ and over $k+1, \dots$, is defined as in [15] so as to satisfy

$$\sum_{i=1}^3 Q_i^{\pi'_{k+1}}(t) \leq \sum_{i=1}^3 Q_i^{\pi'_k}(t), \quad \forall t \geq 0, \quad \forall k \geq 1. \quad (45)$$

Again, by construction, it is clear that

$$\lim_{k \rightarrow \infty} Q_i^{\pi'_k}(t) \stackrel{s}{=} Q_i^{\tilde{\pi}_{IQ}^{(3)}}(t), \quad \forall t \geq 0, \quad 1 \leq i \leq N, \quad (46)$$

where $\stackrel{s}{=}$ means over every sample path (and is stronger than “a.s.”). Eqn. 45 together with Eqn. 46 give us 21. This completes the proof. \square

¹⁶The reader should note that there is a *typo* in [15] that labels two results as Lem 4.1. Here, we refer to the latter as 4.2 to avoid confusion.

G. Proof of Prop. 6

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

$$\begin{aligned}
 P\{S_2 = 1\} &= P\{S_2 = 1 \mid q_1 + q_3 > 0\}P\{q_1 + q_3 > 0\} \\
 &\quad + P\{S_2 = 1 \mid q_1 + q_3 = 0\}P\{q_1 + q_3 = 0\}. \\
 &\leq 0 \cdot P\{q_1 + q_3 > 0\} + 1 \cdot P\{q_1 + q_3 = 0\} \\
 &= 1 - P\{q_1 + q_3 > 0\} \\
 &= 1 - P\{q_1 > 0 \text{ or } q_3 > 0\} \\
 &\leq 1 - P\{A_1(t) > 0 \text{ or } A_3(t) > 0\} \\
 &= P\{A_1(t) = A_3(t) = 0\} = (1 - (0.5 - \epsilon - \delta))^2 \\
 &= (0.5 + \epsilon + \delta)^2
 \end{aligned} \tag{47}$$

If $x := (0.5 + \epsilon) = \lambda_2$, to prove the instability of $\pi_{OQ}^{(3)}$ one only needs to solve the nonlinear program (48) below. That will establish that $P\{S_2(t) = 1\} < \lambda_2$, and hence prove that Queue 2 is unstable.

$$\begin{aligned}
 \text{Find} \quad & (x + \delta)^2 < x, \\
 \text{s.t.} \quad & \delta > 0, \\
 & x > 0.5, \\
 & x < 1, \\
 & x + \delta \leq 1.
 \end{aligned} \tag{48}$$

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

H. Proof of Lem. 11

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 $\zeta(t) \in \{0, 1\}^N$. 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 $\zeta(t) \in \{0, 1\}^{2N-1}$ 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 $\zeta(t) \in \{0, 1\}^{2N-1}$ gets mapped to a unique activation vector $\mathbf{s}(t) \in \{0, 1\}^{2N-1}$, resulting in an admissible policy. \square

I. Proof of Prop. 14

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

Lemma 28. Let A and B be two independent random variables, with A taking values in $\{0, 1, 2, \dots\}$ and B taking values in $\{0, 1\}$. Define $Z := (A - B)^+$. Then,

$$\mathbb{E}Z = \mathbb{E}A - P\{B = 1\}(1 - P\{A = 0\}). \tag{49}$$

PROOF. Let $p_k = P\{A = k\}, k \geq 0$, and $q = P\{B = 1\}$.

Then, $P\{Z = 0\} = P\{A = 0\} + P\{A = 1, B = 1\} = p_0 + p_1q$, and for all $k \geq 1$,

$$\begin{aligned}
 P\{Z = k\} &= P\{A = k, B = 0\} + P\{A = k + 1, B = 1\} \\
 &= p_k(1 - q) + p_{k+1}q.
 \end{aligned} \tag{50}$$

Hence,

$$\begin{aligned}
 \mathbb{E}Z &= \sum_{k=1}^{\infty} kP\{Z = k\} \\
 &= \sum_{k=1}^{\infty} k(p_k(1 - q) + p_{k+1}q) \\
 &= (1 - q) \sum_{k=1}^{\infty} kp_k + q \sum_{k=1}^{\infty} (k + 1 - 1)p_{k+1} \\
 &= (1 - q) \mathbb{E}A + q \sum_{k=1}^{\infty} (k + 1)p_{k+1} - q \sum_{k=1}^{\infty} p_{k+1} \\
 &= (1 - q) \mathbb{E}A + q(\mathbb{E}A - p_1) - q(1 - p_1 - p_0) \\
 &= \mathbb{E}A - q(1 - p_0).
 \end{aligned} \tag{51}$$

■

Lemma 29. Under the policy $\pi_{TD}^{(3)}$,

$$\begin{aligned} \lim_{t \rightarrow \infty} P \{Q_1(t) = 0, Q_2(t) = 0, Q_3(t) = 0\} = \\ (1 - \lambda_1) \left(1 - \frac{\lambda_2}{1 - \lambda_1}\right) \left(1 - \frac{\lambda_3}{1 - \lambda_2}\right). \end{aligned} \quad (52)$$

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. 3) already showed that when Queues 1 and 2 are started out in their steady state distributions,

$$P \{s_3(t) = 1\} = 1 - \lambda_2. \quad (53)$$

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. 28, we see that

$$\begin{aligned} \mathbb{E}Q_3(t+1) &= \mathbb{E}Q_3(t) \\ &\quad - P \{s_3(t) = 1\} (1 - P \{Q_3(t) = 0\}) + \lambda_3, \\ \Rightarrow 1 - P \{Q_3(t) = 0\} &= \frac{\lambda_3}{P \{s_3(t) = 1\}}, \\ \Rightarrow P \{Q_3(t) = 0\} &= 1 - \frac{\lambda_3}{1 - \lambda_2}, \text{ in the steady state.} \end{aligned} \quad (54)$$

Next, note that under $\pi_{TD}^{(3)}$, the system¹⁷ 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,

$$Q(t+1) = Q(t) - \mathbb{I}_{\{Q(t) > 0\}} - \mathbb{I}_{\{Q_1(t) > 0, Q_3(t) > 0\}} + A(t+1). \quad (55)$$

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 $t \rightarrow \infty$, we get

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

■

We now continue with the proof of Prop. 14. 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

¹⁷In this proof, by “system”, we mean the 3 queues system.

- Queues 1, 2 and 3 are all empty.

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.

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

In equality \star , we used the result of Lem. 29. 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. \square

J. 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.

$$\begin{aligned}
P\{Q_3 > 0, Q_1 = 0, Q_2 > 0\} &= P\{Q_3 > 0, Q_2 > 0\} \\
&\quad \times P\{Q_1 = 0 | Q_3 > 0, Q_2 > 0\} \\
&= (1 - \lambda_1) P\{Q_3 > 0, Q_2 > 0\}
\end{aligned} \tag{56}$$

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. 56 we gather that $P\{s_4(t) = 1\} = 1 - \lambda_3$. Hence,

$$\begin{aligned}
P\{s_4(t) = 1\} &= 1 - \lambda_3 \\
&= P\{Q_3 = 0\} \\
&\quad + P\{Q_3 > 0, Q_1 = 0, Q_2 > 0\} \\
&\stackrel{\dagger a}{=} \left(1 - \frac{\lambda_3}{1 - \lambda_2}\right) \\
&\quad + (1 - \lambda_1) P\{Q_3 > 0, Q_2 > 0\} \\
\Rightarrow P\{Q_3 > 0, Q_2 > 0\} &= \frac{1}{1 - \lambda_1} \left(1 - \lambda_3 - \left(1 - \frac{\lambda_3}{1 - \lambda_2}\right)\right) \\
&= \frac{1}{1 - \lambda_1} \lambda_3 \left(\frac{1}{1 - \lambda_2} - 1\right) \\
&= \frac{\lambda_2}{1 - \lambda_1} \cdot \frac{\lambda_3}{1 - \lambda_2} \\
&= P\{Q_2 > 0\} \cdot P\{Q_3 > 0\},
\end{aligned} \tag{57}$$

where equality $\dagger a$ above comes from Eqn. (54). Next,

$$\begin{aligned}
(1 - \lambda_1) \left(1 - \frac{\lambda_2}{1 - \lambda_2}\right) &= P\{Q_1 = 0, Q_2 = 0\} \\
&= P\{Q_1 = 0, Q_2 = 0, Q_3 > 0\} \\
&\quad + P\{Q_1 = 0, Q_2 = 0, Q_3 = 0\} \\
&\stackrel{\dagger b}{=} (1 - \lambda_1) P\{Q_2 = 0, Q_3 > 0\} \\
&\quad + (1 - \lambda_1) \left(1 - \frac{\lambda_2}{1 - \lambda_1}\right) \left(1 - \frac{\lambda_3}{1 - \lambda_2}\right) \\
\Rightarrow P\{Q_2 = 0, Q_3 > 0\} &= \left(1 - \frac{\lambda_2}{1 - \lambda_1}\right) \frac{\lambda_3}{1 - \lambda_2} \\
&= P\{Q_2 = 0\} \cdot P\{Q_3 > 0\},
\end{aligned} \tag{58}$$

where, in equality $\dagger b$ we have made use of Lem. 29. Next,

$$\begin{aligned}
 P\{Q_2 = 0, Q_3 = 0\} &= P\{Q_1 > 0, Q_2 = 0, Q_3 = 0\} \\
 &\quad + P\{Q_1 = 0, Q_2 = 0, Q_3 = 0\} \\
 &= \lambda_1 P\{Q_2 = 0, Q_3 = 0\} \\
 &\quad + (1 - \lambda_1) \left(1 - \frac{\lambda_2}{1 - \lambda_1}\right) \left(1 - \frac{\lambda_3}{1 - \lambda_2}\right),
 \end{aligned} \tag{59}$$

which means that

$$\begin{aligned}
 (1 - \lambda_1) P\{Q_2 = 0, Q_3 = 0\} &= (1 - \lambda_1) \left(1 - \frac{\lambda_2}{1 - \lambda_1}\right) \left(1 - \frac{\lambda_3}{1 - \lambda_2}\right) \\
 \Rightarrow P\{Q_2 = 0, Q_3 = 0\} &= \left(1 - \frac{\lambda_2}{1 - \lambda_1}\right) \left(1 - \frac{\lambda_3}{1 - \lambda_2}\right) \\
 &= P\{Q_2 = 0\} \cdot P\{Q_3 = 0\}.
 \end{aligned} \tag{60}$$

Finally,

$$\begin{aligned}
 P\{Q_2 > 0, Q_3 = 0\} &= 1 - (P\{Q_2 > 0, Q_3 > 0\} \\
 &\quad + P\{Q_2 = 0, Q_3 > 0\}) + P\{Q_2 = 0, Q_3 = 0\} \\
 &= 1 - \left(\frac{\lambda_2}{1 - \lambda_1} \cdot \frac{\lambda_3}{1 - \lambda_2} + \left(1 - \frac{\lambda_2}{1 - \lambda_1}\right) \cdot \frac{\lambda_3}{1 - \lambda_2}\right. \\
 &\quad \left.+ \left(1 - \frac{\lambda_2}{1 - \lambda_1}\right) \left(1 - \frac{\lambda_3}{1 - \lambda_2}\right)\right) \\
 &= 1 - \left(\frac{\lambda_3}{1 - \lambda_2} + 1 - \frac{\lambda_2}{1 - \lambda_1} - \frac{\lambda_3}{1 - \lambda_2}\right. \\
 &\quad \left.+ \frac{\lambda_2}{1 - \lambda_1} \frac{\lambda_3}{1 - \lambda_2}\right) \\
 &= \frac{\lambda_2}{1 - \lambda_1} \left(1 - \frac{\lambda_2}{1 - \lambda_1}\right) \\
 &= P\{Q_2 > 0\} \cdot P\{Q_3 = 0\}.
 \end{aligned} \tag{61}$$

From Eqn. (57), Eqn. (58), Eqn. (60) and Eqn. (61) 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.

K. Proof of Prop. 15

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,

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

$$\begin{aligned}
& +P\{Q_1 = 0, Q_2 > 0, Q_4 = 0\} \\
& +P\{Q_1 = 0, Q_2 = 0, Q_3 = 0, Q_4 = 0\}, \tag{62}
\end{aligned}$$

where in equality c we have used the independence results of Sec. XIV-J. Now, consider the subsystem comprising queues 1, 2 and 4 under this policy and call this subsystem Q_{124} . Since $\pi_{TD}^{(5)}$ restricted to the first 4 queues is essentially $\pi_{TD}^{(4)}$, the top-down policy for the 4 queue system, and since Thm. 14 already showed that $\pi_{TD}^{(4)}$ 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 $\pi_{TD}^{(4)}$ and hence $\pi_{TD}^{(5)}$, 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,

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

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

$$\begin{aligned}
& P\{Q_1(t) = 0, Q_2(t) = 0, Q_3(t) = 0, Q_4(t) > 0\} \\
& + P\{Q_1(t) > 0, Q_3(t) = 0, Q_4(t) > 0\} \\
& + P\{Q_1(t) = 0, Q_2(t) > 0, Q_4(t) > 0\} \\
& = \lambda_1 + \lambda_2 + \lambda_4 - P\{Q_1(t) > 0\} - P\{Q_1(t) = 0, Q_2(t) > 0\} \\
& = \lambda_1 + \lambda_2 + \lambda_4 - \lambda_1 - (1 - \lambda_1) \frac{\lambda_2}{1 - \lambda_1} \\
& = \lambda_4. \tag{63}
\end{aligned}$$

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. 63 on the first three terms on the LHS of the above equation and invoking the independence results in Sec. XIV-J 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. 62, 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 $\pi_{TD}^{(5)}$ and $\pi_{BU}^{(5)}$, the top-down and bottom-up policies are stable. \square

\square

L. Proof of Lem. 17

The proof of throughput-optimality of $\pi_{TI}^{(4)}$ is along the lines of the proof of Thm. 12. By that we mean we shall show that the policy is a splicing of two throughput-optimal scheduling policies for the 3-queue system. Recall from our earlier discussion that Policy $\pi_{TI}^{(4)}$ is defined as follows.

At time t :

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

Notice that the above definition can be split into two parts, the first acting on queues 1, 2 and 3 and the second on queues 2, 3 and 4, as shown below.

At time t :

- 1) Check $[\zeta_1(t), \zeta_2(t), \zeta_3(t)]$
 - a) If $i_2(t) = 1$, $\mathbf{S}(t) = [0, 1, 0, 1]$.
 - b) Else if $i_3(t) = 1$, $\mathbf{S}(t) = [1, 0, 1, 0]$.
 - c) Else $\mathbf{S}(t) = [1, 0, 0, 1]$.
- 2) Check $[\zeta_2(t), \zeta_3(t), \zeta_4(t)]$
 - a) If $i_2(t) = 1$, $\mathbf{S}(t) = [0, 1, 0, 1]$.
 - b) Else if $i_3(t) = 1$, $\mathbf{S}(t) = [1, 0, 1, 0]$.
 - c) Else $\mathbf{S}(t) = [1, 0, 0, 1]$.

The first portion of the policy, i.e., part 1 can be seen as a very simple variant of policy $\pi_{IQ}^{(3)}$, which was the non MSM policy defined in Sec. V-E. Instead of always scheduling queues 1 and 3 when $Q_2(t) = 0$, here, we schedule queues 1 and 3 when $[\zeta_2(t), \zeta_3(t)] = [0, 1]$ and queues 1 and 4 when $Q_2(t) = Q_3(t) = 0$. This amounts to choosing $\mathbf{S}(t) = [1, 0, 1]$ when $Q_2(t) = 0$, but $Q_3(t) > 0$, and $\mathbf{S}(t) = [1, 0, 0]$ when $Q_2(t) = Q_3(t) = 0$. This is permissible since it does not violate the definition of $\pi_{IQ}^{(3)}$ in any way. We already know that $\pi_{IQ}^{(3)}$ is stabilizing for all $[\lambda_1, \lambda_2, \lambda_3] \in \Lambda_3^o$, which is the interior of the capacity region for 3-queue path-graph interference networks defined in (24).

The second portion of the above policy, i.e., part 2 can be seen as a similar variant of policy $\pi_{TD}^{(3)}$ which is defined on queues 2, 3 and 4. From the analysis of the 3-queue system we already know that $\pi_{TD}^{(3)}$ is stabilizing for all $[\lambda_2, \lambda_3, \lambda_4] \in \Lambda_3^o$. Since any $\lambda \in \Lambda_4^o$ satisfies $\lambda_i + \lambda_{i+1} < 1$, $1 \leq i \leq 3$, it automatically means that $[\lambda_1, \lambda_2, \lambda_3] \in \Lambda_3^o$, and $[\lambda_2, \lambda_3, \lambda_4] \in \Lambda_3^o$. The analysis of $\pi_{TI}^{(4)}$ so far implies that the rate region stabilized by it ($\Lambda_{\pi_{TI}^{(4)}}$) satisfies

$$\Lambda_4^o \subseteq \Lambda_{\pi_{TI}^{(4)}}^o, \quad (64)$$

where Λ_4^o is the interior of the capacity region of the system with N queues. But since rate vectors outside Λ_4^o cannot be stabilized anyway, $\pi_{TI}^{(4)}$ is indeed throughput-optimal. \square

M. Proof of Prop. 18

From Table. II, we see that $\tilde{\pi}_3^{(4)}$ differs from $\tilde{\pi}_1^{(4)}$ at exactly one system occupancy vector, viz, $\zeta(t) = [1, 1, 1, 1]$. When $\zeta(t) = [1, 1, 1, 1]$, both activate the same number of queues, but while $\tilde{\pi}_3^{(4)}$ serves queue 1 and 3 while $\tilde{\pi}_1^{(4)}$ serves queues 2 and 4. This means that all four conditions in the proof of delay optimality of the policy $\tilde{\pi}_{IQ}^{(3)}$ (see Sec. XIV-F) are satisfied. Note that condition 4 therein is vacuously satisfied in this case, since $N = 4$ is not an odd number. Hence, using the same technique as in Sec. XIV-F, we see that

$$\sum_{i=1}^4 Q_i^{\tilde{\pi}_3^{(4)}}(t) \leq \sum_{i=1}^4 Q_i^{\tilde{\pi}_1^{(4)}}(t), \quad \forall t \geq 0.$$

But by switching the roles of $\tilde{\pi}_3^{(4)}$ and $\tilde{\pi}_1^{(4)}$ and going through the same argument, we also see that

$$\sum_{i=1}^4 Q_i^{\tilde{\pi}_1^{(4)}}(t) \leq \sum_{i=1}^4 Q_i^{\tilde{\pi}_3^{(4)}}(t), \quad \forall t \geq 0,$$

which leads us to the conclusion that when the initial conditions are identical, both the policies have the same total system occupancy distributions, i.e.,

$$\sum_{i=1}^4 Q_i^{\tilde{\pi}_3^{(4)}}(t) \stackrel{st}{=} \sum_{i=1}^4 Q_i^{\tilde{\pi}_1^{(4)}}(t), \quad \forall t \geq 0. \quad (65)$$

Since $\tilde{\pi}_1^{(4)}$ is strongly stable, arguments following (29), show us that $\tilde{\pi}_3^{(4)}$ is strongly stable as well. Finally, a similar proof with $\tilde{\pi}_2^{(4)}$ and $\tilde{\pi}_4^{(4)}$ shows that

$$\sum_{i=1}^4 Q_i^{\tilde{\pi}_4^{(4)}}(t) \stackrel{st}{=} \sum_{i=1}^4 Q_i^{\tilde{\pi}_2^{(4)}}(t), \quad \forall t \geq 0. \quad (66)$$

and hence, that $\tilde{\pi}_4^{(4)}$ is throughput optimal. This concludes the proof. \square

N. Proof of Prop. 19

Consider rate vectors of the form $\lambda_1 = (\lambda_1, \lambda_2, \lambda_3, 0)$ and $\lambda_2 = (0, \lambda_2, \lambda_3, \lambda_4)$ in Λ^o . First consider λ_1 and let $Q_4(0) = 0$, under this rate vector. By the Borel Cantelli Lemma applied to the Bernoulli arrival process, a zero arrival rate implies that, w.p.1, on sample paths of $A_4(t)$, there will be finitely many arrivals. This means that, w.p.1, along sample paths the policy $\tilde{\pi}_2^{(4)}$ restricted to Queues 1, 2 and 3 reduces to the policy described below. Note that the fourth coordinate of $\mathbf{s}(t)$ is not shown since Queue 4 is always empty.

At time t ,

- 1) If $Q_3(t) > 0$, $\mathbf{s}(t) = [1, 0, 1]$.
- 2) Else, if $Q_2(t) > 0$, $\mathbf{s}(t) = [0, 1, 0]$.
- 3) Else, $\mathbf{s}(t) = [1, 0, 0]$.

This is simply $\pi_{BU}^{(3)}$. It follows that the distribution of sum queue length converges to that of the sum queue length of $\pi_{BU}^{(3)}$. In the same way, $\tilde{\pi}_1^{(4)}$ now reduces to

At time t ,

TABLE V: Comparison of $\mathbf{S}(t)$ under policies in $\Pi_M^{(4)}$ and $\tilde{\Pi}^{(4)}$. Note here, that $s_1 = [1, 0, 1, 0]$, $s_2 = [0, 1, 0, 1]$ and $s_3 = [1, 0, 0, 1]$.

$[i_4(t), i_3(t), i_2(t), i_1(t)]$	$\Pi_M^{(4)}$	$\tilde{\Pi}^{(4)}$
0000	1001	1001
0001	1001	1001
0010	1010	1010
0011	$\{s_1, s_2\}$	1010
0100	0101	0101
0101	0101	0101
0110	$\{s_1, s_2\}$	$\{s_1, s_2\}$
0111	1010	1010
1000	1001	1001
1001	1001	1001
1010	1010	1010
1011	$\{s_1, s_3\}$	1010
1100	$\{s_1, s_2\}$	0101
1101	$\{s_2, s_3\}$	0101
1110	1010	1010
1111	$\{s_1, s_2, s_3\}$	$\{s_1, s_2\}$

- 1) If $\zeta(t) = [1, 1, 1]$, $\mathbf{s}(t) = [1, 0, 1]$.
- 2) Else, if $Q_2(t) > 0$, $\mathbf{s}(t) = [0, 1, 0]$.
- 3) Else, if $Q_3(t) > 0$, $\mathbf{s}(t) = [1, 0, 1]$.
- 4) Else, $\mathbf{s}(t) = [1, 0, 0]$.

This is simply $\tilde{\pi}_{IQ}^{(3)}$, which, as we have already seen, is delay optimal (in the stochastic ordering sense) for the system with 3 queues. In Sec. XII we show arrival rate vectors for which strict inequality holds. Since $\tilde{\pi}_1^{(4)}$ and $\tilde{\pi}_2^{(4)}$ differ from $\tilde{\pi}_3^{(4)}$ and $\tilde{\pi}_4^{(4)}$ respectively only on $\zeta(t) = [1, 1, 1, 1]$, and since this occupancy never occurs (Queue 4 is always empty), the same λ_1 continues to work. This proves Equations (30). A similar argument can be made with λ_2 to prove Eqns. (31). This concludes the proof. \square

O. Proof of Prop. 20

Denote by s_1, s_2 and s_3 the three activation vectors $[1, 0, 1, 0]$, $[0, 1, 0, 1]$ and $[1, 0, 0, 1]$. Any MSM policy in $\Pi_M^{(4)}$ will only choose from among these three vectors in any slot. Consider once again, the set of all 16 possible values that $\zeta(t)$ can take, as given in Table. V. As the table shows, among the occupancy vectors shaded red, $\zeta(t) = [0110]$ is the only one for which policies in both sets completely agree on the set of activation vectors from which to choose. We will, therefore, focus on the other five in the remainder of the proof. Recall that the condition which separates $\Pi_M^{(N)}$ from $\tilde{\Pi}^{(N)}$ is condition 3 in Lem. 1. Since $\pi \in \Pi_M^{(4)} \setminus \tilde{\Pi}^{(4)}$ there exists at least one occupancy vector among the five, for which π chooses an activation vector that violates this condition.

Consider for example, $\zeta(t) = [1, 1, 0, 0]$, which means that $j_1 = 1$ and $j_2 = 2$ and suppose, under this occupancy vector, that π chooses s_2 , i.e., serves queues 1 and 3. We have already seen that Condition 3 in Lem. 1 (that gave the conditions for an activation vector to be MSM) demands that a policy schedule queues 2 and 4, i.e., choose $s_1 = [0, 1, 0, 1]$. Let π' be the policy that $\zeta(t) = [1, 1, 0, 0] \mapsto s_1 = [0, 1, 0, 1]$ and is identical to π for all other occupancy vectors. The objective is to now prove that π' produces stochastically smaller system backlog than π by using the procedure in the proof of delay optimality of $\tilde{\pi}_{IQ}^{(3)}$ (Sec. XIV-F).

Specifically, consider condition 2 in the proof. From the preceding discussion we gather that when $\zeta(t) = [1, 1, 0, 0]$, there exists $l < 4$ such that π' does not serve Queue l while π does serve it. This is obviously Queue 1. But in this case, π' does serve Queue $l + 1$ in the same slot, while π does not, which means that condition 2 is satisfied. Note that since N here is an even number and π and π' are identical on all other occupancy vectors, conditions Conditions 3 and 4 are vacuously satisfied. Hence, following the procedure outlined in the rest of the proof of delay optimality of $\tilde{\pi}_{IQ}^{(3)}$, we see that π' results in stochastically smaller sum queue lengths than π .

The same argument can now be given for any combination of the shaded occupancy vectors (in Table. V) for which policies in $\Pi_M^{(4)} \setminus \tilde{\Pi}^{(4)}$ violate condition 3 in Lem. 1. This proves the result. \square

P. Proof of Prop. 22

We extend our initial idea of Property \mathcal{P} to prove this proposition as follows. For all $1 \leq m \leq N$, 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 $2 \leq m \leq N$,

$$D_1(t) + D_m(t) = 0 \iff Q_1(t) + Q_m(t) = 0, \forall t \geq 0. \quad (67)$$

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

- $Q_1(t) > 0$ 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.

$$L(\mathbf{Q}(t)) := \sum_{m=2}^N (Q_1(t) + Q_m(t))^2 \quad (68)$$

□

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

The proof consists of two parts. We will first prove that $\Phi_{IC}^{(S)}(T)$ 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. 5b).

So consider once again, a *collocated* system of N queues described by a fully connected interference graph. As is the case in Sec. XI-A1, suppose the system only knows $\zeta(t) \in \{0, 1\}^N$, at times $t = 0, T, 2T, \dots$. Following Sec. XI-A1, arrivals in the k^{th} frame are *not* served in the k^{th} frame. We denote by ψ_T , the scheduling policy that, during $kT, kT + 1, \dots, KT + T - 1$, 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 $\{\lambda \in \mathbb{R}_+^N \mid \sum_{i=1}^N \lambda_i < 1\}$. In what follows, we will analyse the process $\{\mathbf{q}(k), k \geq 0\}$, where $\mathbf{q}(k) := \mathbf{Q}(kT)$.

Lemma 30. Under ψ_T , for any λ inside the capacity region,

- the process $\{\mathbf{q}(k), k \geq 0\}$ 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

$$\mathbb{E}_{\psi_T} \sum_{i=1}^N Q_i(kT) \leq \alpha T, \quad \forall k \geq 0. \quad (69)$$

PROOF. Let $A_i[x, y]$ denote the number of arrivals to Queue i over the slots $x, x + 1, \dots, y$. 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

$$q(k + 1) = (q(k) - T)^+ + A(k + 1), \quad (70)$$

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,

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

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

$$\begin{aligned} \mathbb{E}_{\psi_T} [q^2(k + 1) - q^2(k)] &\leq \mathbb{E}_{\psi_T} q^2(k) + (N^2 + 1)T \\ &\quad - 2\epsilon \mathbb{E}_{\psi_T} q(k), \\ \mathbb{E}_{\psi_T} \sum_{i=1}^N Q_i(kT) &\stackrel{*}{\leq} \frac{(N^2 + 1)}{2\epsilon} T, \end{aligned} \quad (71)$$

$$\Rightarrow \limsup_{k \rightarrow \infty} \frac{1}{kT} \sum_{l=0}^{k-1} \mathbb{E}_{\psi_T} \sum_{i=1}^N Q_i(lT) < \infty, \quad (72)$$

In inequality \star , we have used the fact that $\mathbb{E}_{\psi_T} q^2(k+1) \geq 0$. In particular, Eqn. 72 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. (71) 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 $\left[\left(\sum_{j \in C_1} Q_j(t) + \sum_{j \in C_k} Q_j(t) \right) \right]$. It also means that delay with $\Phi^{(T)}$ increases linearly in T . □

R. Proof of Prop. 24

This proof proceeds along the same lines as the proof of delay optimality of Policy $\tilde{\pi}_{IQ}^{(3)}$ that we presented in Sec. XIV-F. $\phi_{IC}^{(S)}$ 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 $\tilde{\pi}_{IQ}^{(3)}$ and will not be repeated. □

S. Proof of Prop. 25

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

$$Q_{1,2}(t+1) = Q_{1,2}(t) - \mathbb{I}_{\{Q_{1,2}(t) > 0\}} + A_{1,2}(t+1), \quad (73)$$

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 (12) 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

$$\begin{aligned} \mathbb{E}Q_{1,2}(t+1) &= \mathbb{E}Q_{1,2}(t) - P\{Q_{1,2}(t) > 0\} + \lambda_1 + \lambda_2, \\ \Rightarrow P\{Q_{1,2}(t) = 0\} &= 1 - \lambda_1 - \lambda_2 \end{aligned} \quad (74)$$

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

$$\begin{aligned} P\{S_{C_3}(t) = 1\} &= \lambda_1 + (1 - \lambda_1 - \lambda_2) \\ &= 1 - \lambda_2 > \lambda_3. \end{aligned} \quad (75)$$

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

T. Proof of Prop. 23

We first show that $\phi_{IC}^{(S)}$ satisfies Eqn. 67 at every $t \geq 0$. Thereafter, the analysis in the proof of $\tilde{\phi}_{IC}^{(S)}$ using the same Lyapunov function as in Eqn. (68) 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 $\phi_{IC}^{(S)}$ 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 XIV-P and proceeds along the same lines. □

U. Proof of Thm. 27

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

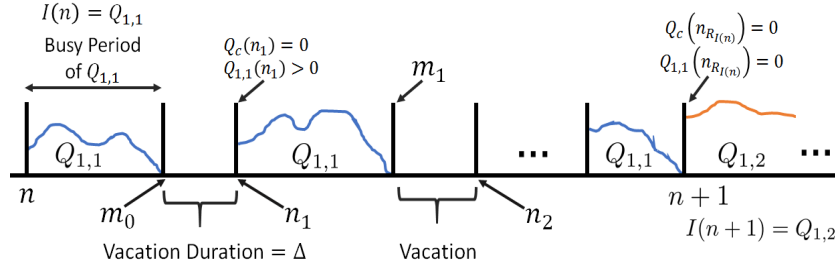


Fig. 18: A sample path illustrating the service process to one of the peripheral queues, here, $Q_{1,1}$. Even with *one* queue in the central clique the resulting service process to peripheral queues is found to be quite complex. Note that here, $I(n) = Q_{1,1}$, in Clique 1, while $I(n+1) = Q_{1,2}$ since it was chosen in Step 2b in the definition of $\phi_{CS}^{(S)}$.

a) *Service Processes under $\phi_{CS}^{(S)}$* :: 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, \dots, 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. 18 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. 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. 18, $Q_{1,1}$ 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. 18 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 analysing $\phi_{CS}^{(S)}$ 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. 18). 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, $n_0 = n$.

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 $\rho = \sum_{j=1}^{N_1} \rho_j$ the total load on the clique. The interference constraints dictate that $\rho < 1$. Let $b = \mathbb{E}B = \frac{1}{1-\lambda_c}$. 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

$$\mathbb{E}\Delta = \sum_{k=0}^{\infty} k \lambda_c^k (1 - \lambda_c) = \frac{\lambda_c}{1 - \lambda_c} \quad (76)$$

$$\begin{aligned} \mathbb{E}A_j(\Delta) &= \mathbb{E}(\mathbb{E}[A_j(\Delta) | \Delta]) = \mathbb{E}(\lambda_j \mathbb{E}\Delta) \\ &= \lambda_c \frac{\lambda_j}{1 - \lambda_c} = \lambda_c \rho_j \end{aligned} \quad (77)$$

Now, notice that

$$Q_j(n_1) = \begin{cases} Q_j(n) + \underbrace{A_j(G_{I(n)}(Q_{I(n)}(n)))}_{\text{arrivals to Queue } j \text{ during busy period of Queue } I(n)} \\ \quad + \underbrace{A_j(\Delta)}_{\text{arrivals during the subsequent vacation}}, & \text{for } j \neq I(n) \\ A_j(\Delta), & \text{for } j = I(n). \end{cases} \quad (78)$$

Using (77) and (78), we get

$$\begin{aligned} b\mathbb{E}[Q_j(n_1)|\mathbf{Q}(n)] &\leq bQ_j(n) + bQ_{I(n)}(n) \frac{\rho_j}{1 - \rho_{I(n)}} + b\lambda_c \rho_j \\ \mathbb{E}\left[\sum_{j=1}^{N_1} bQ_j(n_1)|\mathbf{Q}(n)\right] &\leq \sum_{j=1}^{N_1} bQ_j(n) + \left(\frac{\rho - \rho_j}{1 - \rho_{I(n)}} - 1\right) bQ_{I(n)}(n) \\ &\quad + b\lambda_c \rho \\ &\stackrel{\star}{=} \sum_{j=1}^{N_1} bQ_j(n) + \underbrace{h_{I(n)}(\rho - 1) bQ_{I(n)}(n)}_{\text{strictly negative}} \\ &\quad + b\lambda_c \rho, \end{aligned} \quad (79)$$

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 (79), $(\rho - 1)$ is *strictly negative*, by capacity constraints. Similarly,

$$\begin{aligned} \mathbb{E}\left[\sum_{j=1}^{N_1} bQ_j(n_2)|\mathbf{Q}(n)\right] &= \mathbb{E}\left[\underbrace{\mathbb{E}\left[\sum_{j=1}^{N_1} bQ_j(n_2)|\mathbf{Q}(n_1), \mathbf{Q}(n)\right]}_{\sigma(\mathbf{Q}(n)) \subset \sigma(\mathbf{Q}(n_1), \mathbf{Q}(n))} \middle| \mathbf{Q}(n)\right] \\ &\stackrel{\dagger}{=} \mathbb{E}\left[\mathbb{E}\left[\sum_{j=1}^{N_1} bQ_j(n_2)|\mathbf{Q}(n_1)\right] \middle| \mathbf{Q}(n)\right] \\ \mathbb{E}\left[\sum_{j=1}^{N_1} bQ_j(n_2)|\mathbf{Q}(n)\right] &\stackrel{\star 1}{\leq} \mathbb{E}\left[\sum_{j=1}^{N_1} bQ_j(n_1)|\mathbf{Q}(n)\right] \\ &\quad + h_{I(n)}(\rho - 1) b\mathbb{E}\left[Q_{I(n)}(n_1)|\mathbf{Q}(n)\right] + b\lambda_c \rho \\ &= \left(\sum_{j=1}^{N_1} bQ_j(n) + h_{I(n)}(\rho - 1) bQ_{I(n)}(n) + b\lambda_c \rho\right) \\ &\quad + (h_{I(n)}(\rho - 1) b\lambda_c \rho_{I(n)}) + b\lambda_c \rho, \\ &= \sum_{j=1}^{N_1} bQ_j(n) + 2b\lambda_c \rho \\ &\quad + \underbrace{(\rho - 1) b h_{I(n)} (Q_{I(n)} + \rho_{I(n)})}_{\text{strictly negative}}. \end{aligned}$$

Proceeding similarly,

$$\begin{aligned} \mathbb{E}\left[\sum_{j=1}^{N_1} bQ_j(n_k)|\mathbf{Q}(n)\right] &= \sum_{j=1}^{N_1} bQ_j(n) + kb\lambda_c \rho \\ &\quad + (\rho - 1) b h_{I(n)} (Q_{I(n)} \\ &\quad + (k - 1)\rho_{I(n)}), \quad \forall k \geq 1. \end{aligned} \quad (80)$$

Equality \dagger follows from the Markovian nature of the evolution of the queue-length vector, and we have used (79) 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 (80) 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 $R_{I(n)}$.

$$\begin{aligned} P(R_{I(n)} = k) &= \prod_{l=1}^k P(A_{I(n)}^{(l)}(\Delta) > 0) \\ &\quad \times P(A_{I(n)}^{(k+1)}(\Delta) = 0), \quad \forall k \geq 0. \end{aligned} \quad (81)$$

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

$$\begin{aligned} P(A_{I(n)}^{(l)}(\Delta) = 0) &= \sum_{k=0}^{\infty} P(A_{I(n)}^{(l)}(\Delta) = 0 | \Delta = k) \\ &= \sum_{k=0}^{\infty} (1 - \lambda_{I(n)})^k \lambda_c (1 - \lambda_c)^k \\ &= \frac{1 - \lambda_c}{1 - (1 - \lambda_{I(n)})\lambda_c}, \end{aligned} \quad (82)$$

Which gives us

$$\mathbb{E}R_{I(n)} = \frac{\lambda_c \lambda_{I(n)}}{1 - \lambda_c} + 1 \quad (83)$$

Now, using (83) in (80), we get

$$\begin{aligned} \mathbb{E} \left[\sum_{j=1}^{N_1} bQ_j(n_{R_{I(n)}}) | \mathbf{Q}(n) \right] &= \sum_{j=1}^{N_1} bQ_j(n) + \left(\frac{\lambda_c \lambda_{I(n)}}{1 - \lambda_c} + 1 \right) b\lambda_c \rho \\ &\quad + (\rho - 1) bh_{I(n)} \\ &\quad \times \left(Q_{I(n)} + \left(\frac{\lambda_c \lambda_{I(n)}}{1 - \lambda_c} \right) \rho_{I(n)} \right), \end{aligned}$$

from which we get the conditional drift as

$$\begin{aligned} &\mathbb{E} \left[\sum_{j=1}^{N_1} bQ_j(n+1) - \sum_{j=1}^{N_1} bQ_j(n) | \mathbf{Q}(n) \right] \\ &= \mathbb{E} \left[\sum_{j=1}^{N_1} bQ_j(n_{R_{I(n)}}) - \sum_{j=1}^{N_1} bQ_j(n) | \mathbf{Q}(n) \right] \\ &\leq \left(\frac{\lambda_c \lambda_{I(n)}}{1 - \lambda_c} + 1 \right) b\lambda_c \rho \\ &\quad + (\rho - 1) bh_{I(n)} \left(Q_{I(n)} + \left(\frac{\lambda_c \lambda_{I(n)}}{1 - \lambda_c} \right) \rho_{I(n)} \right) \\ &< -\epsilon, \end{aligned}$$

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 N_1 DTMCs $\{\mathbf{Q}_{nN_1+K}\}_{n=0}^{\infty}$ is positive recurrent for $K = 0, 1, \dots, N_1 - 1$. 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. Proof of Prop. XI-D

Let $A_j(s, t] := \sum_{k=s+1}^t A_j(k)$ be the total number of arrivals to C_j in $\{s + 1, s + 2, \dots, t\}$. Recall that the total backlog of clique j at the beginning of slot t is denoted by $Q_j(t) = \sum_{k=1}^{N_j} Q_{j,k}(t)$. We will prove that the process $\{\mathbf{Q}(kT) := [Q_1(kT), Q_2(kT), Q_3(kT)], t \geq 0\}$ is stable. For this, we first look at a policy that is clearly suboptimal (in terms of delay) compared to $\psi_{1,T}^{(3)}$. This policy only serves, during the k^{th} frame, the packets that have already arrived to a scheduled queue at or before the beginning of the frame, i.e., it does not serve the packets arriving at the queue over $(kT, KT + (T - 1)]$.

Firstly, since C_1 is served whenever it is non empty, $Q_1(t)$ evolves as $Q_1((k+1)T) = Q_1(kT) - \mathbb{I}_{\{Q_1(kT) > 0\}} + A_1(kT, (k+1)T]$. Also notice that if either C_1 or C_2 is non empty at the beginning of a frame, one of these two cliques always gets served, i.e.,

$$Q_{1,2}((k+1)T) = Q_{1,2}(kT) - T\mathbb{I}_{\{Q_{1,2}(kT) > 0\}} + A_{1,2}(kT, kT + (T-1)] \quad (84)$$

Using the modified Property \mathcal{P} for batch arrivals and departures, we see that $\{Q_{1,2}(kT), k = 0, 1, 2, \dots\}$ is strongly stable, and, also being an aperiodic, irreducible DTMC, it is also positive recurrent. This proves the existence of a stationary measure $\mu_{1,2}$ for the chain. Using arguments similar to the one in (74), we see that, in the steady state, $P\{Q_{1,2}(kT) = 0\} = \mu_{1,2}(\{0\}) = T(1 - \lambda_1 - \lambda_2)$. Now, coming to C_3 , let $S_{C_3}(kT, kT + (T-1)]$ be the total number of slots in the k^{th} frame during which clique 3 is offered service. Then,

$$Q_3((k+1)T) = \left(Q_3(kT) - S_{C_3}(kT, kT + (T-1)] \right)^+ + A_3(kT, kT + (T-1)] \quad (85)$$

Assume $Q_{1,2}(0) \sim \mu_{1,2}$, then

$$\begin{aligned} P\{S_{C_3}(t) = 1\} &= P\{Q_1(kT) = 1\} + P\{Q_{1,2}(kT) = 0\} \\ &= \lambda_1 T + T(1 - \lambda_1 - \lambda_2) \\ &= (1 - \lambda_2)T \\ &> \lambda_3 T = \mathbb{E}A_3(kT, kT + (T-1)]. \end{aligned} \quad (86)$$

Hence, Clique 3 is also stable and this proves the claim. \square

W. Simulation Details

Here we provide the arrival rate vectors for:

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

REFERENCES

- [1] M. Honrubia, "Industrial iot is booming thanks to a drop in sensor prices," 2017, <https://www.ennomotive.com/industrial-iot-sensor-prices/>.
- [2] M. Raza, N. Aslam, H. Le-Minh, S. Hussain, Y. Cao, and N. M. Khan, "A critical analysis of research potential, challenges and future directives in industrial wireless sensor networks," *IEEE Communications Surveys & Tutorials*, 2017.
- [3] 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.
- [4] E. Dahlman, S. Parkvall, and J. Skold, *4G: LTE/LTE-advanced for mobile broadband*. Academic press, 2013.
- [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] 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.
- [7] 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.
- [8] 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.
- [9] 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.
- [10] 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.
- [11] 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.
- [12] 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.
- [13] 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.

- [14] 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.
- [15] 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.
- [16] T. Ji, E. Athanasopoulou, and R. Srikant, "On optimal scheduling algorithms for small generalized switches," *IEEE/ACM Transactions on Networking*, vol. 18, no. 5, pp. 1585–1598, 2010.
- [17] A. Taghavi, A. Vem, J.-F. Chamberland, and K. R. Narayanan, "On the design of universal schemes for massive uncoordinated multiple access," in *Information Theory (ISIT), 2016 IEEE International Symposium on*. IEEE, 2016, pp. 345–349.
- [18] K. R. Narayanan and H. D. Pfister, "Iterative collision resolution for slotted aloha: An optimal uncoordinated transmission policy," in *Turbo Codes and Iterative Information Processing (ITC), 2012 7th International Symposium on*. IEEE, 2012, pp. 136–139.
- [19] 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.
- [20] 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.
- [21] D. Shah *et al.*, "Heavy traffic analysis of optimal scheduling algorithms for switched networks," *Annals of Applied Probability*, 2007.
- [22] 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.
- [23] J. L. Gross and J. Yellen, *Graph theory and its applications*. CRC press, 2005.
- [24] R. Diestel, "Graph theory. 2005," *Grad. Texts in Math*, vol. 101, 2005.
- [25] S. T. Maguluri, R. Srikant, and L. Ying, "Heavy traffic optimal resource allocation algorithms for cloud computing clusters," *Arxiv preprint arXiv:1206.1264*, 2012.
- [26] 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.
- [27] 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.
- [28] 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.
- [29] A. Warrior, 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.
- [30] S. T. Maguluri, "Optimal scheduling algorithms for ad hoc wireless networks," 2011.
- [31] 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.
- [32] I. Rhee, A. Warrior, 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.
- [33] P. Kinney, "The 802.15.4 CCA method," *Project: IEEE P802.15 Working Group for Wireless Personal Area Networks (WPANs)*, Submitted: Nov 14, 2001.
- [34] 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.
- [35] 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.
- [36] 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>.
- [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 M.Tech. and PhD degrees from the Indian Institute of Technology (IIT) Madras and the Indian Institute of Science (IISc) Bangalore, respectively. He is currently a postdoctoral fellow with the Reinforcement Learning Research Laboratory ((RL)² Lab) at the Technion, Israel Institute of Technology. His research interests include analysis of pricing in deregulated electricity markets, stochastic control and reinforcement learning and resource allocation in wireless communication networks.

Aditya Gopalan 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.