Optimal Control for Wireless Networks

by

Abhishek Sinha

Submitted to the Department of Aeronautics and Astronautics
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

June 2017

Author .....
Department of Aeronautics and Astronautics
June

Certified by .....
Eytan Modiano
Professor
Thesis Supervisor

Certified by .....
Leandros Tassiulas
Professor
Thesis Committee Member

Certified by .....
David Gamarnik
Professor
Thesis Committee Member

Accepted by .....
Hamsa Balakrishnan
Associate Professor
Chair, Graduate Program Committee

Optimal Control for Wireless Networks

by

Abhishek Sinha

Submitted to the Department of Aeronautics and Astronautics
on June, in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy

Abstract

One of the most fundamental problems in Computer Networking is to efficiently route packets belonging to different sessions, such as unicast, broadcast, multicast and anycast, collectively known as the generalized flows. The goal of this thesis is to design an efficient routing, and wireless link scheduling policy, that maximizes net throughput. Currently, the only known throughput-optimal policy is the *Backpressure* policy for the unicast problem. In this thesis, we propose provably optimal algorithms for the broadcast and the generalized flow problems.

Our study begins with the problem of optimal broadcasting in a wireless Directed Acyclic Graph (DAG). Existing policies achieve the broadcast capacity by balancing traffic over a set of spanning trees, which are difficult to maintain in a large and time-varying network. We propose a fundamentally new broadcast policy, which is decentralized, utilizes local information only, does not require the use of global topological structures, such as spanning trees. It also yields a new and computationally efficient characterization of the broadcast capacity in wireless DAGs. We next study the problem of broadcasting in networks with arbitrary topology and derive a new dynamic broadcast policy which can be viewed as "Backpressure on sets". This yields an efficient solution to the problem when combined with a multi-class in-order packet scheduling rule.

Finally, we study the generalized flow problem and derive an online dynamic policy, called Universal Max-Weight (UMW). To the best of our knowledge, UMW is the first throughput-optimal algorithm of such versatility in this context. Conceptually, the UMW policy is derived by relaxing the precedence constraints associated with multi-hop routing and then solving a min-cost routing and max-weight scheduling problem on a virtual network of queues. When specialized to the unicast setting, unlike Backpressure, the UMW policy yields a throughput-optimal cycle-free routing and link scheduling policy. The proposed algorithmic paradigm is surprisingly general and can be used to solve other related problems, such as optimal broadcasting in wireless networks with point-to-multipoint links. The proof of throughput-optimality of the UMW policy combines techniques from stochastic Lyapunov theory with a sample path argument from adversarial queueing theory and may be of independent

theoretical interest.

Thesis Supervisor: Eytan Modiano
Title: Professor

Thesis Committee Member: Leandros Tassiulas
Title: Professor

Thesis Committee Member: David Gamarnik
Title: Professor

असतो मा साद गमय, तमसो मा ज्योतिर् गमय, मृत्योर मा अमृतम् गमय

(Lead us from ignorance to truth, Lead us from darkness to light, Lead us from death to deathlessness)

- Brihadaranyaka Upanishad ($\sim$ 9th to 6th century BCE)

Dedicated to my mother, Purnima Sinha, and my father, Subhash Sinha

Acknowledgements [1]

Firstly, I would like to express my deep gratitude towards my advisor, Prof. Eytan Modiano, for guiding me throughout my PhD journey. He taught me the art of identifying fundamental research problems that have both engineering utility and academic merit. He emphasized the importance of having an intuitive understanding of the problem at hand, as well as the ability to back up the intuition with mathematical rigor. I immensely enjoyed the regular weekly meetings with Eytan, where we brainstormed together to identify possible scopes for making progress. Moreover, he taught me the art of perseverance, which was extremely helpful for tackling my dissertation problem, which appeared to be quite intractable at the very beginning.

I am grateful to Prof. Leandros Tassiulas and Prof. David Gamarnik for having them on my thesis committee. My short visit to Yale University in 2015 and working with Prof. Tassiulas proved to be instrumental in tackling the broadcasting problem in time-varying wireless networks. My early collaborations with Georgios Paschos and Chih-Ping Li proved to be extremely valuable for making headway on my dissertation problem. I would also like to extend my gratitude towards my Master's thesis advisor, Prof. Anurag Kumar from the Indian Institute of Science (IISc.), Bangalore, for introducing me to the fascinating field of networks and probability.

Special thanks to the Communication and Networking Research group (CNRG) members for making my stay at MIT memorable. I immensely enjoyed our day-to-day conversation with Jianan and Anurag, discussing open problems with Thomas, collaborating (and playing pool!) with Igor, and hanging out with Rajat, who has

[1]Research partially supported by DARPA I2O and Raytheon BBN Technologies under Contract No. HROO II-I 5-C-0097, NSF under grants CNS-1217048, CNS-1524317, AST-1547331, ONR under grant N00014-12-1-0064, and ARO MURI under grant W911NF-08-1-0238.

been my friend since my IISc days. I would also like to thank the students and faculties of the Laboratory for Information and Decision Systems (LIDS) for creating a vibrant intellectual environment and setting the highest standard for fundamental research.

Finally, I would like to thank my parents, for their never-failing love, support, and encouragement. They taught me that nothing is impossible if one has the highest work ethics, genuineness in purpose, and faith in own ability.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 The Generalized Network Flow Problem

The *Generalized Network Flow* (**GNF**) problem refers to the efficient transportation of packets, generated at source node(s), to a set of designated destination node(s) over a multi-hop wired or wireless network. Depending on the number of destination nodes associated with each source node, the **GNF** problem encompasses several fundamental networking problems including *unicast* (single destination node), *broadcast* (all node are destination nodes), *multicast* (some nodes are destination nodes), and *anycast* (several choices for a single destination node) problem.

Over the last few decades, tremendous amount of research efforts has been directed towards addressing each of the above problems in different networking contexts. One of the most prominent *unicast* algorithm in this context is *Backpressure* [1], which is *throughput-optimal* for unicast flows, under fairly general assumptions. Numerous extensions of the original *Backpressure* algorithm have also been proposed in the literature. Some of the works relevant to this thesis will be discussed in the literature review section 1.3.

In general, the design of efficient policies for the GNF problem faces several major challenges. Unlike the unicast case, the lack of *flow conservation* property in the broadcast and multicast problem makes the use of traditional queueing-theoretical approaches difficult. Wireless channels suffer from interference, and a broadcast policy

needs to activate non-interfering links at every time slot. Wireless network topologies undergo frequent changes, so that packet forwarding decisions must be made in an adaptive fashion. Some dynamic algorithms have been proposed in the literaure [2], that balance traffic over several spanning trees. However, these algorithms are not suitable for wireless networks because enumerating all spanning trees is computationally prohibitive, more so when this is to be done repeatedly as and when the topology changes with time.

Moreover, despite the increasingly diverse mix of internet traffic, to the best of our knowledge, there exists no universal solution to the general problem, only isolated solutions that do not interoperate and are often suboptimal. The main difficulty in designing an efficient dynamic algorithm stems from packet duplications involved in the broadcast or multicast problem, which invariably leads to an exponential number of states in the natural formulation of the problem.

In this thesis, our objective is to comprehensively study the *Generalized Network Flow* (**GNF**) problem, with a special attention to the Broadcast problem, in particular. In Chapters 2, 3, 4 and 6 we will study the **Broadcast problem**, which is a special case of the **GNF** problem, and a fundamentally important flow-problem in its own right. Then, in Chapter 5 we will study the **GNF** problem in its full generality, and propose a general algorithmic paradigm, called **Universal Max-Weight** (**UMW**), that solves the problem. The **UMW**policy is based on the intuitive idea of a *precedence-relaxed* network, which is a surprisingly general principle, and is applicable to a wide array of other related flow problems. As an example, the same idea of precedence-relaxation will be used in Chapter 6, to solve the throughput-optimal Broadcast problem in wireless networks with point-to-multipoint links.

1.2 System Model and Problem Formulation

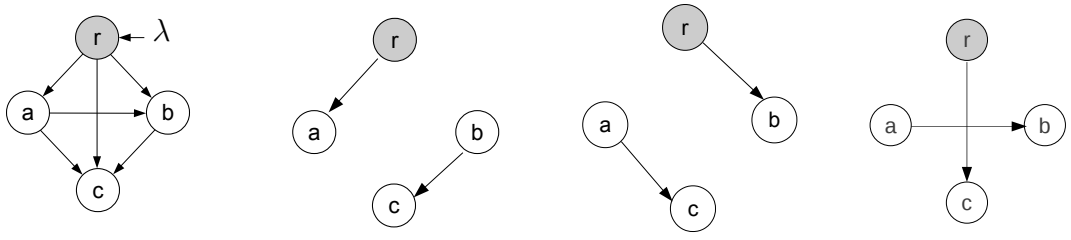
In the following, we outline the general set up which we will be considering throughout the thesis. In addition to this, we will make specific assumptions applicable to each individual chapter. Such additional assumptions will be stated at the beginning of

each chapter.

1.2.1 Network Model

We consider a wireless network with arbitrary topology, represented by the graph $\mathcal{G}(V, E)$. The network consists of $|V| = n$ nodes and $|E| = m$ links [1]. Time is slotted. A link (i, j), if activated, can transmit c_{ij} packets per slot. Due to the wireless interference constraints, only certain subsets of links may be activated together at any slot. The set of all admissible link activations is known as the *activation set* and is denoted by $\mathcal{M} \subseteq 2^E$. We do not impose any restriction on the structure of the activation set $\mathcal{M}$. As an example, in the case of *node-exclusive* or primary interference constraint [3], the activation set $\mathcal{M}_{\text{primary}}$ consists of the set of all *matchings* [4] in the graph $\mathcal{G}(V, E)$. Wired networks are a special case of the above model, where the activation set $\mathcal{M}_{\text{wired}} = 2^E$. In other words, in wired networks, packets can be transmitted over all links simultaneously. See Figure 1-1 for an example of a wireless network with primary interference constraints.

For simplicity of exposition, unless stated otherwise, the network topology will be assumed to be static. The problem of efficient broadcasting in a time-varying DAG network will be undertaken in Chapter 3.



(a) a wireless network (b) activation vector $\boldsymbol{s}_1$ (c) activation vector $\boldsymbol{s}_2$ (d) activation vector $\boldsymbol{s}_3$

Figure 1-1: A wireless network and its three feasible link activations under the primary interference constraint.

[1] In general, the links will be assumed to be *point-to-point*. The problem of broadcasting with *point-to-multipoint* links will be undertaken in Chapter 6.

1.2.2 Traffic Model

The set of all distinct classes of incoming traffic is denoted by $\mathcal{C}$. A class c traffic is identified by its source node $s^{(c)} \in V$ and the set of its required destination nodes $\mathcal{D}^{(c)} \subseteq V$. As explained below, by varying the structure of the destination set $\mathcal{D}^{(c)}$ of class c, this general framework yields the following four fundamental flow problems as special cases:

- **Unicast**: All class c packets, arriving at a source node $s^{(c)}$, are required to be delivered to a single destination node $\mathcal{D}^{(c)} = \{t^{(c)}\}$.
- **Broadcast**: All class c packets, arriving at a source node $s^{(c)}$, are required to be delivered to all nodes in the network, i.e., $\mathcal{D}^{(c)} = V$.
- **Multicast**: All class c packets, arriving at a source node $s^{(c)}$, are required to be delivered to a proper subset of nodes $\mathcal{D}^{(c)} = \{t_1^{(c)}, t_2^{(c)}, \ldots, t_k^{(c)}\} \subsetneq V$.
- **Anycast**: A Packet of class c, arriving at a source node $s^{(c)}$, is required to be delivered to *any one* of a given set of k nodes $\mathcal{D}^{(c)} = t_1^{(c)} \oplus t_2^{(c)} \oplus \ldots \oplus t_k^{(c)}$. Thus the anycast problem is similar to the unicast problem, with all destinations forming a single *super* destination node.

Arrivals are i.i.d. at every slot, with $A^{(c)}(t)$ packets from class c arriving at the source node $s^{(c)}$ at slot t. The mean rate of arrival for class c is $\mathbb{E}A^{(c)}(t) = \lambda^{(c)}$. The arrival rate to the network is characterized by the vector $\boldsymbol{\lambda} = \{\lambda^{(c)}, c \in \mathcal{C}\}$. The total number of external packet arrivals to the entire network at any slot t is assumed to be bounded by a finite number $A_{\max}$.

1.2.3 Policy-Space

An admissible policy π for the generalized network flow problem executes the following two actions at every slot t:

- LINK ACTIVATIONS: Activating a subset of interference-free links $\boldsymbol{s}(t)$ from the activation set $\mathcal{M}$.
- PACKET DUPLICATIONS AND FORWARDING: Possibly duplicating [2] and forwarding packets over the activated links. Due to the link capacity constraint, at most one packet may be transmitted over an active link per slot.

The set of all admissible policies is denoted by Π. The set Π is unconstrained otherwise and includes policies which may use all past and future packet arrival information.

A policy $\pi \in \Pi$ is said to *support an arrival rate-vector* $\boldsymbol{\lambda}$ if, under the action of the policy π, the destination nodes of any class c receive distinct class c packets at the rate $\lambda^{(c)}, c \in \mathcal{C}$. Formally, let $R^{(c)}(t)$ denote the number of distinct class-c packets, received in common by all destination nodes $i \in \mathcal{D}^{(c)}$ [3], under the action of the policy π, up to time t.

Definition 1.2.1 [Policy Supporting Rate-Vector $\boldsymbol{\lambda}$]*: A policy $\pi \in \Pi$ is said to support an arrival rate vector $\boldsymbol{\lambda}$ if*

$$\liminf_{t\to\infty} \frac{R^{(c)}(t)}{t} = \lambda^{(c)}, \quad \forall c \in \mathcal{C}, \text{ w.p.1} \tag{1.1}$$

The network-layer capacity region $\boldsymbol{\Lambda}(\mathcal{G}, \mathcal{C})$ [4] is defined to be the set of all support-

[3] To be precise, the *super*-destination node in case of Anycast.

[4] Note that, Network-layer capacity region is, in general (e.g. multicast), different from the Information-Theoretic capacity region [5].

able rates, i.e.,

$$\mathbf{\Lambda}(\mathcal{G},\mathcal{C}) \stackrel{\text{def}}{=} \{\boldsymbol{\lambda} \in \mathbb{R}_{+}^{|\mathcal{C}|} : \exists \pi \in \Pi \text{ supporting } \boldsymbol{\lambda}\} \tag{1.2}$$

Clearly, the set $\mathbf{\Lambda}(\mathcal{G},\mathcal{C})$ is *convex* (using the usual *time-sharing* argument). A policy $\pi^* \in \Pi$, which supports any arrival rate $\boldsymbol{\lambda}$ in the interior of the capacity region $\mathbf{\Lambda}(\mathcal{G},\mathcal{C})$, is called a *throughput-optimal* policy. Roughly speaking, our focus in this thesis is solving the following fundamental problem:

Problem 1.2.2 *Design a distributed, low-complexity, dynamic, throughout-optimal policy for wireless networks.*

1.3 Literature Review

In the **Broadcast** problem, packets generated at a source need to be distributed among all nodes in the network. For efficient broadcasting, one needs to use appropriate packet replication and forwarding to eliminate redundant transmissions. This is especially important in power-constrained wireless networks which suffer from interference and packet collisions. Broadcast applications include mission-critical military communications [6], live video streaming [7], information dissemination in vehicular networks [8], file searching [9], in-network function computation [10], and data dissemination in sensor networks [11] among others.

In the literature, a simple method for wireless broadcast is to use packet flooding [12]. The flooding approach, however, leads to redundant transmissions and collisions, known as *broadcast storm* [13]. In the classic paper of Edmonds [14], the broadcast capacity of a wired network is characterized and an algorithm is proposed to compute the maximum number of edge-disjoint spanning trees, which together achieve the maximum broadcast throughput. The algorithm in [14] is combinatorial in nature and does not have a wireless counterpart, with associated interference-free edge

activations. Following Edmonds' work, a variety of different broadcast algorithms have been proposed in the literature, each one targeted to optimize different metrics such as delay [15], number of retransmissions [16], energy consumption [17] and fault-tolerance [18]. In the context of optimizing throughput, [19] proposes a randomized packet forwarding policy, which is shown to be optimal for wired networks via fluid limit techniques. However, extending this algorithm to the wireless setting proves to be difficult.

Another fundamental feature of the wireless medium is the inherent point-to-mutipoint nature of wireless links, where a packet transmitted by a node can be heard by all its neighbors. This feature, also known as the wireless broadcast advantage [20], is especially useful in broadcast applications. Additionally, because of inter-node interference, the set of simultaneous transmissions in a wireless network is restricted to the set of non-interfering feasible schedules only. The problem of designing throughput optimal broadcast policy in wireless networks with *point-to-multi-point links* was considered in [21], where the authors studied a highly restrictive "scheduling-free" model, where it is assumed that scheduling decisions are made by a central controller, acting independently of their algorithm. With this assumption, they used the randomized packet forwarding scheme of [19], to design a randomized wireless broadcast algorithm. This algorithm was proved to be throughput optimal *with respect to the given schedule*, using fluid limit techniques.

The **Multicast** problem is a generalization of the broadcast problem, in which the packets generated at a source node needs to be efficiently distributed to a subset of nodes in the network. In its combinatorial version, the multicast problem reduces to finding the maximum number of edge-disjoint trees, spanning the source node and destination nodes. This problem is known as the *Steiner Tree Packing* problem, which is NP-hard [22]. Numerous algorithms have been proposed in the literature for solving the multicast problem. In [2] [23], back-pressure type algorithms are proposed for multicasting over wired and wireless networks respectively. These algorithms forward packet over a set of pre-computed distribution trees, and are limited to the throughput obtainable by these trees. Moreover, computing and maintaining these

trees is impractical in large and time-varying networks. We note that because of the need for packet duplications, the Multicast and Broadcast problems do not satisfy standard flow conservation constraints, and thus the design of throughput-optimal algorithms is non-trivial.

The **Unicast** problem involves a single source and a single destination. The celebrated Back-Pressure (BP) algorithm [1] was proposed for the unicast problem. In this algorithm, the routing and scheduling decisions are taken based on local queue length differences. As a result, BP explores all possible paths for routing, induces routing loops, and usually takes a long time for convergence, resulting in considerable latency, especially in lightly loaded networks [24]. Subsequently, a number of refinements have been proposed to improve the delay characteristics of the BP algorithm. In [25], the BP algorithm is combined with hop length based shortest path routing for faster route discovery, [26] proposes a variant with shadow queues, [27] suggests incorporating a bias term in the backpressure calculations, and [28] proposes a second order variant utilizing the Hessian matrix to improve delay. However, as reported in [24] based on real testbed implementation, most of these algorithms do not perform well in practice.

The **Anycast** problem is a generalization of the Unicast problem and it involves routing from a single source to *any one* of the *several* given destinations. Anycast is increasingly used in Content-Distribution Networks (CDNs) for optimally distributing geo-replicated contents [29]. The paper [30] proposes a simple variant of the back-pressure routing scheme for the anycast problem. However, the above limitations of the BP scheme still remains as in the Unicast problem.

1.4 Thesis Contributions

1.4.1 Throughput-Optimal Broadcast Policy for Wireless DAGs

In Chapter 2 we consider the problem of throughput-optimal broadcasting in static wireless DAGs, and in Chapter 3 we extend the results to time-varying wireless DAGs.

Our major contributions in these two chapters include the following:

1. A fully decentralized efficient dynamic Max-Weight Broadcast policy, which is provably throughput-optimal. For designing this policy, we introduced a cleverly-designed restricted policy-space Π^*, which follows the constraint of *in-order* packet delivery. We showed how the underlying DAG property may be exploited to design an optimal policy with only a *linear* number of state-variables (as compared to the *exponential* number of state-variables in the natural formulation of the broadcast problem.)

2. We derived a new characterization of the broadcast capacity λ^* of a wireless DAG. This is expressed as a LP with exponentially number of constraints. Moreover, under the primary interference constraint, we show that the LP can be solved in polynomial time.

3. We extend the above characterization to time-varying wireless DAG networks. Moreover, we obtain tight upper and lower bounds for the broadcast capacity for time-varying DAG networks with general interference constraints.

1.4.2 Throughput-Optimal Broadcast Policy for Arbitrary Networks

In Chapter 4 we design a throughput-optimal broadcast policy for arbitrary networks. Our major contributions in this chapter includes the following:

1. We show a nice parallel between the proposed throughput-optimal broadcast policy π^* and the well-known throughput-optimal policy for the unicast problem, namely the *backpressure* policy. More precisely, the broadcast policy π^* may be thought of as performing "*backpressure on sets*", as opposed to backpressure on nodes.

2. Since the policy π^* has high computational complexity, we extend the *in-order* policy space of Chapter 2 and propose a *multi-class heuristic* broadcast policy

with a tunable number of classes. We also derive several structural properties of the multi-class policy space. The heuristic policy is empirically shown to perform well with small number of classes.

1.4.3 Optimal Control for the Generalized Network Flow Problem

In Chapter 5 we propose a general algorithmic paradigm for solving the **GNF** problem. Our contributions in this chapter includes the following:

1. We introduce the notion of *precedence-relaxed* virtual network, which is much easier to analyze and control, than the original multi-hop network.

2. Using Lyapunov-drift theory, we devise a throughput-optimal routing and scheduling policy (called **UMW**) for the virtual network for different kinds of flows. In brief, the routing policy reduces to a suitable *weighted shortest-route* problem and the link-scheduling policy reduces to a Max-Weight activation problem, where the links are weighted by the corresponding *virtual queue lengths*.

3. Using tools from adversarial queuing theory, we show that the above routing and scheduling policy may be combined with an appropriate packet scheduling policy, such that the overall policy is throughput-optimal, in the sense that the physical queues are *stable* for all admissible packet arrival rates.

4. When specialized to the unicast setting, the **UMW**policy yields a very different and much attractive throughput-optimal control policy than the original Backpressure policy. In particular, it involves *source routing* (route of a packet is fixed at the source on its arrival), yields less queuing delay (due to always routing along *acyclic* routes) and incurs much less queuing complexity (only one physical queue per link, *irrespective* of the number and types of flows in the network). We expect **UMW**to play a decisive role in the emerging networking technologies, such as, Software Defined Networks [31], [32].

1.4.4 Throughput-Optimal Broadcast Policy for Wireless Networks with Point-to-Multipoint Links

In Chapter 6 we consider the problem of throughput-optimal broadcast in wireless networks with point-to-multipoint wireless links. Our major contributions in this chapter are as follows:

1. We extend the technique of *virtual network*, introduced in Chapter 5, to tackle this problem. This establishes the generality and efficacy of the virtual network methodology. Interestingly, unlike in Chapter 5, where virtual queues were associated with the edges of the network, in this chapter, the virtual queues correspond to the nodes in the network.

2. We show that the combinatorial version of the wireless Broadcast problem with point-to-multipoint links is **NP**-complete, even in wired setting with network topology being restricted to DAGs. This result should be contrasted with our results in Chapters 2 and 3 about the efficient solubility of the Broadcast problem in wireless DAGs with point-to-point links.

3. We introduce a new Max-Weight control policy and a proof technique by combining the stochastic Lyapunov drift theory with the deterministic adversarial queueing theory. As in Chapter 5, this essentially enables us to derive a stabilizing control policy for a multi-hop network by solving the problem on a simpler precedence-relaxed virtual single-hop network.

Chapter 2

Throughput-Optimal Broadcast on Static Wireless DAGs

2.1 Overview of the Results

In this chapter, we design a throughput-optimal broadcast algorithm for wireless networks whose underlying topology is restricted to be a Directed Acyclic Graph (DAG). To design the algorithm, we start out by considering a rich class of scheduling policies Π that perform arbitrary link activations and packet forwarding. We define the broadcast capacity λ^* as the maximum common rate achievable over this policy class Π. We next enforce two constraints that lead to a tractable set of policies without any loss of throughput-optimality. First, we consider the subclass of policies $\Pi^{\text{in-order}} \subset \Pi$ which delivers packets to all nodes, in the same order they arrive at the source, i.e., *in-order*. Second, we focus on the subset of policies $\Pi^* \subset \Pi^{\text{in-order}}$ that allows the reception of a packet by a node only if all its incoming neighbours have received the packet. It is intuitively apparent that the policies in the more structured class Π^* are easier to describe and analyze, but might not be throughput-optimal. We prove the surprising result that when the underlying network topology is a directed acyclic graph (DAG), there is a broadcast policy $\pi^* \in \Pi^*$ that achieves the broadcast capacity of the network. In contrast, we also find a (non-DAG) network containing a directed cycle in which *no* control policy in the space $\Pi^{\text{in-order}}$ can provably achieve

the broadcast capacity.

To design the optimal broadcast policy π^*, we first establish a *queue-like dynamics* for the system-state, represented by relative packet deficits. This is non-trivial for the broadcast problem because explicit queueing structures are difficult to define due to packet duplications and consequent loss of work-conservation. We subsequently show that, the problem of achieving the broadcast capacity in a DAG reduces to the problem of finding a scheduling policy *stabilizing* the relative packet deficits, which can be solved by utilizing Lyapunov drift analysis techniques [1, 33].

The main contributions of this chapter are as follows:

- We derive a simple characterization of the broadcast capacity of a wireless DAG network. We also show that, for a wired DAG with integral capacities, its broadcast capacity is determined by its minimum in-degree, which is also equal to the maximum number of edge-disjoint directed spanning trees rooted at the source.

- We design a dynamic algorithm that utilizes local queue-length information to achieve the broadcast capacity of a wireless DAG network. This algorithm does not rely on constructing spanning trees and is computationallly efficient. This algorithm also yields a constructive proof of a version of Edmonds' disjoint tree-packing theorem [14], generalized to wireless activations but specialized to DAG topology.

- Based on our characterization of the broadcast capacity, we derive a polynomial-time algorithm to compute the broadcast capacity of any wireless DAG under the primary interference constraints.

- We propose a randomized multiclass extension of the proposed broadcast algorithm, which can be effectively used to do broadcast on wireless networks with arbitrary topology.

- We demonstrate the superior delay performance of our algorithm, as compared

to the centralized tree-based algorithms [2], via numerical simulations. We also explore the efficiency/complexity trade-off of the proposed multiclass extension through extensive numerical simulations.

The rest of the chapter is organized as follows. In Section 3.3, we define the broadcast capacity of a wireless network and provide a useful outer bound on the capacity from a cut-set consideration. In Section 2.3, we propose a dynamic broadcast algorithm that achieves the broadcast capacity in a DAG. In section 2.4, we propose an efficient algorithm for computing the broadcast capacity of any wireless DAG under the primary interference constraints. Our DAG-broadcast algorithm is extended to networks with arbitrary topology in section 2.5. Illustrative simulation results are presented in Section 2.6. Finally, we conclude this chapter in section 2.7.

2.2 The Wireless Broadcast Capacity

Intuitively, the network supports a broadcast rate λ if there exists a scheduling policy under which all network nodes can receive distinct packets at rate λ. The broadcast capacity is the maximally supportable broadcast rate in the network. Formally, we consider a class Π of scheduling policies where each policy $\pi \in \Pi$ consists of a sequence of actions $\{\pi_t\}_{t\geq 1}$ executed at every slot t. Each action π_t comprises of two operations: (i) the scheduler activates a subset of links by choosing a feasible activation vector $\boldsymbol{s}(t) \in \mathcal{S}$; (ii) each node i forwards a subset of packets (possibly empty) to node j over an activated link $e = (i, j)$ (with $\boldsymbol{s}_e(t) = 1$), subject to the link capacity constraint. The policy class Π includes policies that may use all past and future information, and may forward any subset of packets over a link.

Let $R_i^\pi(t)$ be the number of distinct packets received by node $i \in V$ from the beginning of time up to time t, under the action of a policy $\pi \in \Pi$. The time average $\liminf_{T\to\infty} R_i^\pi(T)/T$ is the rate of distinct packets received at node i.

Definition 2.2.1 (Broadcast Policy) *A policy $\pi \in \Pi$ is called a "broadcast policy*

of rate λ" if all nodes receive distinct packets at rate λ, i.e.,

$$\min_{i \in V} \liminf_{T \to \infty} \frac{1}{T} R_i^{\pi}(T) = \lambda, \quad w.\ p.\ 1, \tag{2.1}$$

where λ is the packet arrival rate at the source node $\boldsymbol{r}$.

Definition 2.2.2 *The broadcast capacity λ^* of a wireless network is the supremum of all arrival rates λ for which there exists a broadcast policy $\pi \in \Pi$ of rate λ.*

2.2.1 An upper bound on broadcast capacity λ^*

We characterize the broadcast capacity λ^* of a wireless network by proving a useful upper bound. This upper bound is understood as a necessary cut-set bound of an associated edge-capacitated graph that reflects the time-averaged behaviour of the scheduling policies in Π. We first give an intuitive explanation of the bound, assuming that the limits involved exist. In the proof of Theorem 2.2.3 we rigorously establish the result without this assumption.

Fix a policy $\pi \in \Pi$. Let β_e^{π} be the fraction of time link $e \in E$ is activated under π; i.e., define the vector

$$\boldsymbol{\beta}^{\pi} = (\beta_e^{\pi}, e \in E) = \lim_{T \to \infty} \frac{1}{T} \sum_{t=1}^{T} \boldsymbol{s}^{\pi}(t), \tag{2.2}$$

where $\boldsymbol{s}^{\pi}(t)$ is the chosen link-activation vector by policy π in slot t. The average packet flow rate over a link e under the policy π is upper bounded by the product of the link capacity and the fraction of time the link e is activated, i.e., $c_e \beta_e^{\pi}$. Hence, we can define an associated edge-capacitated graph, $\widehat{\mathcal{G}}^{\pi} = (V, E, (\widehat{c}_e))$ where each link $e \in E$ has capacity $\widehat{c}_e = c_e \beta_e^{\pi}$; see Fig. 2-1 for an example of such an edge-capacitated graph. Next, we provide a bound on the broadcast capacity by maximizing the broadcast capacity on the ensemble of graphs $\widehat{\mathcal{G}}^{\pi}$ over all feasible average edge-activation vectors $\boldsymbol{\beta}^{\pi}$.

Define a *proper cut* U of the network graph $\widehat{\mathcal{G}}^{\pi}$ as a proper subset of the node set

V that contains the source node $\mathtt{r}$. Define the edge-cut E_U associated U as

$$E_U = \{(i,j) \in E \mid i \in U,\ j \notin U\}. \tag{2.3}$$

Since $U \subset V$, there exists a node $n \in V \setminus U$. Consider the throughput (rate of packet reception) of node n under policy π. The max-flow min-cut theorem shows that the throughput of node n cannot exceed the total link capacity $\sum_{e \in E_U} c_e \, \beta_e^\pi$ across the cut U. This cut-set bound is valid even when we consider the general flow of *information* in the network (see Theorem 15.10.1 of [34]). Hence the cut-set bound holds even when we allow network coding operations. By definition of achievable broadcast rate λ^π, we have $\lambda^\pi \leq \sum_{e \in E_U} c_e \, \beta_e^\pi$. This inequality holds for all proper cuts U and we have

$$\lambda^\pi \leq \min_{U\text{: a proper cut}} \sum_{e \in E_U} c_e \, \beta_e^\pi. \tag{2.4}$$

Equation (2.4) holds for any policy $\pi \in \Pi$. Thus, the broadcast capacity λ^* of the wireless network satisfies

$$\begin{aligned} \lambda^* = \sup_{\pi \in \Pi} \lambda^\pi \qquad &\leq \sup_{\pi \in \Pi} \min_{U\text{: a proper cut}} \textstyle\sum_{e \in E_U} c_e \, \beta_e^\pi \\ &\leq \max_{\boldsymbol{\beta} \in \mathrm{conv}(\mathcal{S})} \min_{U\text{: a proper cut}} \textstyle\sum_{e \in E_U} c_e \, \beta_e, \end{aligned}$$

where the last inequality holds because the vector $\boldsymbol{\beta}^\pi$ lies in the convex hull of the activation set $\mathcal{S}$; Refer to Eqn. (2.2). Our first theorem formalizes the above intuitive characterization of the broadcast capacity λ^* of a wireless network.

Theorem 2.2.3 *The broadcast capacity λ^* of a wireless network $\mathcal{G}(V, E, \boldsymbol{c})$ with activation set $\mathcal{S}$ is upper bounded as follows:*

$$\lambda^* \leq \max_{\boldsymbol{\beta} \in \mathrm{conv}(\mathcal{S})} \left(\min_{U\text{: a proper cut}} \sum_{e \in E_U} c_e \, \beta_e \right). \tag{2.5}$$

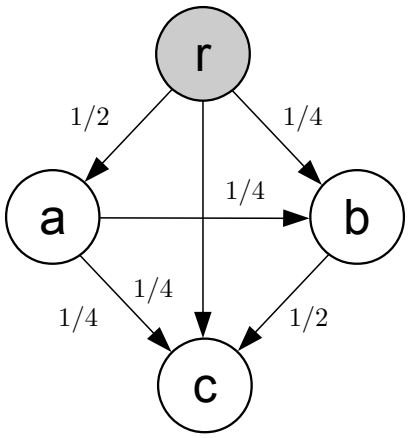


Figure 2-1: The edge-capacitated graph $\widehat{\mathcal{G}}^{\pi}$ for the wireless network with unit link capacities in Fig. 1-1 and under the time-average vector $\boldsymbol{\beta}^{\pi} = (1/2, 1/4, 1/4)$. The link weights are the capacities $c_e\beta_e^{\pi}$. The minimum proper cut in this graph has value $1/2$ (when $U = \{r, a, c\}$ or $\{r, b, c\}$). An upper bound on the broadcast capacity is obtained by maximizing this value over all vectors $\boldsymbol{\beta}^{\pi} \in \text{conv}\,(\mathcal{S})$.

Proof See Appendix 3.3.1.

2.2.2 Constrained Policy-Space: In-order packet delivery

Studying the performance of any arbitrary broadcast policy $\pi \in \Pi$ is analytically formidable because packets are replicated across the network and may be received out of order. To avoid unnecessary re-transmissions, each node must keep track of the identity of the received set of packets, which complicates the system state; because instead of the number of packets received (as in classical back-pressure algorithm [1]), the system state is properly described here by the identity of the subset of packets received at each of the nodes.

To simplify the state, we focus on the subset of policies $\Pi^{\text{in-order}} \subset \Pi$ that enforce the following constraint. It will be shown subsequently that, this restriction can be made without loss of throughput-optimality in a DAG.

Index the packets serially $\{1, 2, \ldots\}$ according to their order of arrival at the source.

Constraint 2.2.4 (In-order packet delivery $\Pi^{\text{in-order}}$) *In this policy-space, a node is allowed to receive a packet p at slot t only if all previous packets $\{1, 2, \ldots, p-1\}$ have been received by that node by slot t.*

In-order packet delivery is practically useful in live media streaming applications [7] where buffering out-of-order packets incurs increased delay and degrades the playback quality. As shown below, Constraint 1 greatly simplifies the state space representation of the system.

Let $R_i(t)$ be the number of distinct packets received by node i by time t. For policies in $\Pi^{\text{in-order}}$, the set of received packets by time t at node i is $\{1, \ldots, R_i(t)\}$. Therefore, the network state in slot t is given by the vector $\boldsymbol{R}(t) = \big(R_i(t), i \in V\big)$.

In section 2.3 we show the existence of a throughput-optimal broadcast policy in the space $\Pi^{\text{in-order}}$ when the underlying topology is a DAG. On the other hand, the following complementary result, Lemma 2.2.5, says that there exists a non-DAG network in which *any* broadcast policy in the space $\Pi^{\text{in-order}}$ is *not* throughput optimal. This implies that the policy-space $\Pi^{\text{in-order}}$ can not, in general, be utilized beyond DAGs while preserving throughput optimality.

Lemma 2.2.5 *Let ${\lambda^*}_{\text{in}-\text{order}}$ be the broadcast capacity of the policy subclass $\Pi^{\text{in}-\text{order}} \subset \Pi$ that enforces in-order packet delivery. There exists a network topology containing a directed cycle such that ${\lambda^*}_{\text{in}-\text{order}} < \lambda^*$.*

Proof See Appendix 2.9.

We will return to the problem of broadcasting in networks with arbitrary topology in Section 2.5, where we will show how the proposed algorithm may be extended to handle the general case.

2.2.3 Achieving the broadcast capacity in a DAG

At this point we concentrate our attention to Directed Acyclic Graphs (DAGs). Graphs in this class are appealing for our analysis because they possess the well-known topological ordering of the nodes [4]. For DAGs, the upper bound (2.5) on the broadcast capacity λ^* in Theorem 2.2.3 will be relaxed further. For each receiver node $v \neq \mathtt{r}$, consider the proper cut U_v that separates the node v from the rest of the network. i.e.,

$$U_v = V \setminus \{v\}. \tag{2.6}$$

Using these collection of cuts $\{U_v, v \neq \mathtt{r}\}$, we obtain a relaxed upper bound λ_{DAG} on the broadcast capacity λ^* as:

$$\begin{aligned} \lambda_{\text{DAG}} &\triangleq \max_{\boldsymbol{\beta} \in \text{conv}(\mathcal{S})} \min_{\{U_v, v \neq \mathtt{r}\}} \sum_{e \in E_{U_v}} c_e \, \beta_e \\ &\geq \max_{\boldsymbol{\beta} \in \text{conv}(\mathcal{S})} \min_{U\text{: a proper cut}} \sum_{e \in E_U} c_e \, \beta_e \geq \lambda^*, \end{aligned} \tag{2.7}$$

where the first inequality uses the subset relation $\{U_v, v \neq \mathtt{r}\} \subseteq \{U\text{: a proper cut}\}$ and the second inequality follows from Theorem 2.2.3. In Section 2.3, we will propose a dynamic policy that belongs to the policy class $\Pi^{\text{in-order}}$ and achieves the broadcast rate λ_{DAG}. Combining this result with (2.7), we establish that the broadcast capacity of a DAG is given by

$$\begin{aligned} \lambda^* = \lambda_{\text{DAG}} &= \max_{\boldsymbol{\beta} \in \text{conv}(\mathcal{S})} \min_{\{U_v, v \neq \mathtt{r}\}} \sum_{e \in E_{U_v}} c_e \, \beta_e, \\ &= \max_{\boldsymbol{\beta} \in \text{conv}(\mathcal{S})} \min_{U\text{: a proper cut}} \sum_{e \in E_U} c_e \, \beta_e. \end{aligned} \tag{2.8}$$

The capacity is achieved by a broadcast policy that uses in-order packet delivery. In other words, we show that imposing the in-order packet delivery constraint does not reduce the broadcast capacity when the topology is a DAG. As a corollary, we also retrieve the result that network-coding operations do not increase the broadcast-capacity in our setting.

From a computational point of view, the equality in Eqn. (2.8) is attractive, because it implies that for computing the broadcast capacity of any wireless DAG, it is enough to consider only those cuts that separate a single (non-source) node from the rest of the network. Note that, there are only $|V| - 1$ of such cuts, in contrast with the total number of cuts in Eqn. (2.5), which is exponential in the size of the network. This fact will be exploited in section 2.4 to develop a strongly poly-time algorithm for computing the broadcast capacity of any wireless DAG network under the primary interference constraints.

2.3 Efficient Broadcast Policy for a DAG

In this section we design a throughput-optimal broadcast policy for wireless DAGs. We start by imposing an additional constraint on packet-forwarding that leads to a new subclass of policies $\Pi^* \subseteq \Pi^{\text{in-order}}$. As we will see, policies in Π^* can be described in terms of relative packet deficits which constitute a simple dynamics. We analyze the dynamics of the minimum relative packet deficits, which behaves like *virtual queues*. We design a dynamic control policy $\pi^* \in \Pi^*$ that stabilizes the virtual queues. The main result of this section is to show that this control policy achieves the broadcast capacity whenever the network topology is a DAG.

2.3.1 System-state by means of packet deficits

We showed earlier in Section 2.2.2 that, constrained to the policy-space $\Pi^{\text{in-order}}$, the system-state is completely represented by the vector $\boldsymbol{R}(t)$. However this constrained policy-class alone is not sufficient to obtain a one-step dynamics of the system, which is an essential prerequisite to design a stabilizing control policy. As a result, we restrict our attention to a sub-class of policies in $\Pi^{\text{in-order}}$, defined as follows.

A node i is called an *in-neighbor* of node j if there exists a directed link $(i, j) \in E$ in the underlying graph $\mathcal{G}$. The set of all in-neighbours of a node j is denoted by $\delta_{\text{in}}(j)$. The out-neighbours of a node is defined similarly.

Constraint 2.3.1 (Policy-space Π^*) *A packet p is eligible for transmission to node j at a slot t, only if all the in-neighbours of j have received packet p in some previous slot.*

We denote this new policy-class by $\Pi^* \subseteq \Pi^{\text{in-order}}$. It will be shown subsequently that this restriction can be done without loss of throughput-optimality. Fig. 2-2 shows the relationship among different policy classes[1].
Following two properties of the system-state $\boldsymbol{R}(t)$ under the action of a policy $\pi \in \Pi^*$ will be useful.

Lemma 2.3.2 *Under any policy $\pi \in \Pi^*$, we have:*

(1) $R_j(t) \leq \min_{i \in \delta_{\text{in}}(j)} R_i(t)$

(2) The indices of packets p that are eligible to be transmitted to the node j at slot t are given by

$$\big\{p \mid R_j(t) + 1 \leq p \leq \min_{i \in \delta_{\text{in}}(j)} R_i(t)\big\}.$$

The proof of the above lemma follows directly from the definition of the policy-space Π^*.

Define the *packet-deficit* $Q_{ij}(t)$ over the link $(i,j) \in E$ to be $Q_{ij}(t) \stackrel{\text{def}}{=} R_i(t) - R_j(t)$. Under a policy in Π^*, $Q_{ij}(t)$ is always non-negative because, by part (1) of Lemma 2.3.2, we have

$$Q_{ij}(t) = R_i(t) - R_j(t) \geq \min_{k \in \delta_{\text{in}}(j)} R_k(t) - R_j(t) \geq 0.$$

The variable $Q_{ij}(t)$ denotes the number of packets received by node i but not by node j upto time t. Intuitively, if all packet-deficits $\{Q_{ij}(t)\}$ are bounded asymptotically, the

[1]We note that, if the network contains a directed cycle, then a deadlock might occur under a policy in Π^* and may yield zero broadcast throughput. However, this problem does not arise when the underlying topology is a DAG.

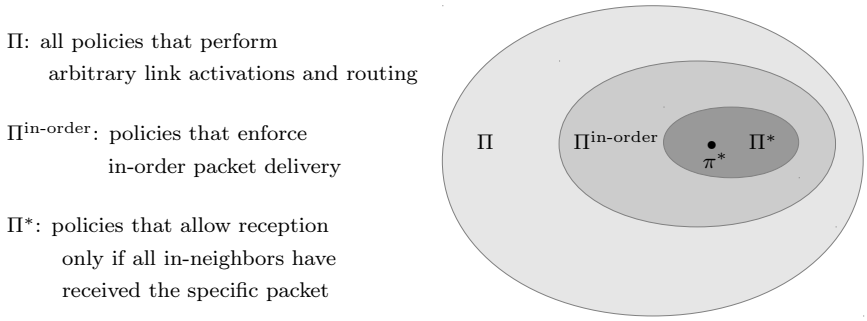


Figure 2-2: Containment relationships among different policy classes.

total number of packets received by any node is not lagging far from the total number of packets generated at the source; hence, the broadcast throughput will be equal to the packet generation rate.

To analyze the system dynamics under a policy in Π^*, it is useful to define the *minimum packet deficit* at node $j \neq r$ by

$$X_j(t) \stackrel{\text{def}}{=} \min_{i \in \delta_{\text{in}}(j)} Q_{ij}(t). \tag{2.9}$$

From part (2) of Lemma 2.3.2, $X_j(t)$ is the maximum number of packets that node j is allowed to receive from its in-neighbors at slot t under Π^*.

As an example, Fig. 3-2 shows that the packet deficits at node j, relative to its in-neighbors a, b, and c, are $Q_{aj}(t) = 8$, $Q_{bj}(t) = 5$, and $Q_{cj}(t) = 4$ respectively. Thus $X_j(t) = 4$ and node j is only allowed to receive four packets in slot t due to Constraint 2.3.1.

We can rewrite $X_j(t)$ as

$$X_j(t) = Q_{i_t^* j}(t), \quad \text{where } i_t^* = \arg \min_{i \in \delta_{\text{in}}(j)} Q_{ij}(t), \tag{2.10}$$

i.e., the node i_t^* is the in-neighbor of node j from which node j has the smallest packet deficit in slot t; ties are broken arbitrarily in deciding i_t^*.[2] Our optimal broadcast

[2]We note that the minimizer i_t^* is a function of the node j and the time slot t and should be properly denoted as $i_t^*(j)$; we slightly abuse the notation by dropping the symbol j from i_t^* throughout to simplify notations.

policy will be described in terms of the minimum packet deficits $\{X_j(t)\}_{j\neq \mathtt{r}}$.

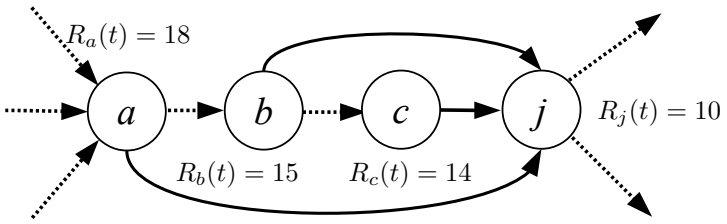


Figure 2-3: Under a policy $\pi \in \Pi^*$, the set of packets available for transmission to node j in slot t is $\{11, 12, 13, 14\}$, which are present at all in-neighbors of the node j. The in-neighbor of j having the smallest packet deficit is $i_t^* = c$, and $X_j(t) = \min\{Q_{aj}(t), Q_{bj}(t), Q_{cj}(t)\} = 4$.

2.3.2 The dynamics of the state variables $\{X_j(t)\}_{j\neq \mathtt{r}}$

We now analyze the dynamics of the state variables

$$X_j(t) = Q_{i_t^* j}(t) = R_{i_t^*}(t) - R_j(t) \tag{2.11}$$

under a policy $\pi \in \Pi^*$. Define the service rate vector $\boldsymbol{\mu}(t) = (\mu_{ij}(t))_{(i,j)\in E}$ by

$$\mu_{ij}(t) = \begin{cases} c_{ij} & \text{if } (i,j) \in E \text{ and the link } (i,j) \text{ is activated,} \\ 0 & \text{otherwise.} \end{cases}$$

Equivalently, we may write $\mu_{ij}(t) = c_{ij} s_{ij}(t)$, where $\boldsymbol{s}(t)$ is the link-activation vector $\boldsymbol{s}(t)$, chosen for slot t. At node j, the increase in the value of number of packets received, i.e., $R_j(t)$, depends on the identity of the received packets; in particular for efficiency, the node j must receive distinct packets. Next, we clarify which set of packets are allowed to be received by node j at time t.

The number of available packets for reception at node j is $\min\{X_j(t), \sum_{k\in V} \mu_{kj}(t)\}$. This is because: (i) $X_j(t)$ is the maximum number of packets node j can receive from its in-neighbours subject to the Constraint 2.3.1; (ii) $\sum_{k\in V} \mu_{kj}(t)$ is the total incoming transmission rate at node j under a given link-activation decision. To correctly derive the dynamics of $R_j(t)$, we consider the following efficiency requirement on policies in Π^*:

Constraint 2.3.3 (Efficient forwarding) *Given a service rate vector $\boldsymbol{\mu}(t)$, node j pulls from the activated incoming links the following subset of packets (denoted by their indices)*

$$\Big\{p \mid R_j(t) + 1 \leq p \leq R_j(t) + \min\{X_j(t), \sum_{k \in V} \mu_{kj}(t)\}\Big\}, \tag{2.12}$$

The specific subset of packets that are pulled over each incoming link are disjoint but otherwise arbitrary.[3]

Constraint 2.3.3 requires that scheduling policies must avoid forwarding the same packet to a node over two different incoming links. Under certain interference models such as the primary interference model, at most one incoming link per node is activated in a slot and Constraint 2.3.3 is redundant.

In Eqn. (2.11), the packet deficit $Q_{i_t^* j}(t)$ increases with $R_{i_t^*}(t)$ and decreases with $R_j(t)$, where $R_{i_t^*}(t)$ and $R_j(t)$ are both non-decreasing. Hence, we can upper-bound the increase of $Q_{i_t^* j}(t)$ by the total service rate of the activated incoming links at node i_t^*, i.e., $\sum_{m \in V} \mu_{m i_t^*}(t)$. Also, we can express the decrement of $Q_{i_t^* j}(t)$ by the exact number of distinct packets received by node j from its in-neighbours, given by $\min\{X_j(t), \sum_{k \in V} \mu_{kj}(t)\}$ by Constraint 2.3.3. Consequently, the one-slot evolution of the variable $Q_{i_t^* j}(t)$ is given by[4]

$$\begin{aligned} Q_{i_t^* j}(t+1) \; &\leq \big(Q_{i_t^* j}(t) - \sum_{k \in V} \mu_{kj}(t)\big)^+ + \sum_{m \in V} \mu_{m i_t^*}(t) \\ &= \big(X_j(t) - \sum_{k \in V} \mu_{kj}(t)\big)^+ + \sum_{m \in V} \mu_{m i_t^*}(t), \end{aligned} \tag{2.13}$$

where $(x)^+ = \max(x, 0)$ and we recall that $X_j(t) = Q_{i_t^* j}(t)$. It follows that $X_j(t)$

[3]Due to Constraints 2.2.4 and 2.3.1, the packets in (2.12) have been received by all in-neighbors of node j.

[4]We emphasize that for a given node j, the node i_t^*, as defined in (2.10), depends on time t and may be different from the node i_{t+1}^*.

evolves over slot t according to

$$X_j(t+1) \overset{(a)}{=} \min_{i \in \delta_{\text{in}}(j)} Q_{ij}(t+1) \overset{(b)}{\leq} Q_{i_t^* j}(t+1)$$
$$\overset{(c)}{\leq} \big(X_j(t) - \sum_{k \in V} \mu_{kj}(t)\big)^+ + \sum_{m \in V} \mu_{m i_t^*}(t), \tag{2.14}$$

where the equality (a) follows the definition of $X_j(t)$, inequality (b) follows because node $i_t^* \in \delta_{\text{in}}(j)$ and inequality (c) follows from Eqn. (2.13). In Eqn. (2.14), if $i_t^* = \mathtt{r}$, the notation is slightly abused to define $\sum_{m \in V} \mu_{mr}(t) = A(t)$ for the source node $\mathtt{r}$, where $A(t)$ is the number of exogenous packet arrival at source at slot t.

2.3.3 A Throughput-optimal Broadcast Policy

Like the Back-Pressure algorithm [1], our broadcast policy is designed to keep the deficit process $\{\boldsymbol{X}(t)\}_{t \geq 0}$ stochastically stable. For this, we regard the variables $X_j(t)$ as virtual queues that follow the dynamics (2.14). By performing drift analysis on the virtual queues $X_j(t)$, we propose the following max-weight-type broadcast policy π^*, described in Algorithm 1. However, the way the weights are computed in π^* (2.16), is very much different from the Back-Pressure algorithm. Also the fundamental feature of packet duplications is essentially new here. The policy π^* belongs to the space Π^* and enforces the constraints 2.2.4, 2.3.1, and 2.3.3. We will show that this policy achieves the broadcast capacity λ^* of a wireless network over the general policy class Π when the underlying topology is a DAG. The steps of the algorithm are illustrated in Fig. 2-4.

Distributed Implementation As evident from the description of Algorithm 1, computation of the weight-vectors $\boldsymbol{W}(t)$ and packet forwarding decisions are made locally by individual nodes. The only network-wide operation that the algorithm needs to perform is step 3, where it needs to select the maximum-weighted feasible activation set. The problem of scheduling the Max-weight activation set in a distributed fashion has been studied extensively in the literature [35] [36]. In particular,

the work of Bui et. al. [36] designs a distributed algorithm for solving the Max-weight scheduling problem with constant overhead in the primary interference setting.

Algorithm 1 Optimal Broadcast Policy π^* for a Wireless DAG:

At each slot t, the network-controller observes the state-variables $\{R_j(t), j \in V\}$ and executes the following actions

1: For each link $(i,j) \in E$, compute the deficit $Q_{ij}(t) = R_i(t) - R_j(t)$ and the set of nodes $K_j(t) \subset \delta_{\text{out}}(j)$ for which node j is their deficit minimizer, i.e.,

$$K_j(t) \leftarrow \big\{k \in V \mid j = \arg\min_{m \in \delta_{\text{in}}(k)} Q_{mk}(t)\big\}. \tag{2.15}$$

The ties are broken arbitrarily (e.g., in favor of the highest indexed node) in finding the $\arg\min(\cdot)$ in Eqn.(2.15).

2: Compute $X_j(t) = \min_{i \in \delta_{\text{in}}(j)} Q_{ij}(t)$ for $j \neq \mathtt{r}$ and assign to link (i,j) the weight

$$W_{ij}(t) \leftarrow \big(X_j(t) - \sum_{k \in K_j(t)} X_k(t)\big). \tag{2.16}$$

3: In slot t, choose the link-activation vector $\boldsymbol{s}(t) = (s_e(t), e \in E)$ such that

$$\boldsymbol{s}(t) \in \arg\max_{\boldsymbol{s} \in \mathcal{S}} \sum_{e \in E} c_e s_e W_e(t). \tag{2.17}$$

4: Every node $j \neq \mathtt{r}$ uses activated incoming links to pull packets $\{R_j(t) + 1, \ldots, R_j(t) + \min\{\sum_i c_{ij} s_{ij}(t), X_j(t)\}\}$ from its in-neighbors according to the Constraint 2.3.3.

5: The vector $(R_j(t), j \in V)$ is updated as follows:

$$R_j(t+1) \leftarrow \begin{cases} R_j(t) + A(t), & j = \mathtt{r}, \\ R_j(t) + \min\{\sum_i c_{ij} s_{ij}(t), X_j(t)\}, & j \neq \mathtt{r}, \end{cases}$$

The next theorem demonstrates the optimality of the broadcast policy π^* described above.

Theorem 2.3.4 *If the underlying topology of $\mathcal{G}$ is a DAG, then for any exogenous packet arrival rate $\lambda < \lambda_{\text{DAG}}$, the broadcast policy π^* yields*

$$\min_{i \in V} \liminf_{T \to \infty} \frac{R_i^{\pi^*}(T)}{T} = \lambda, \quad \textit{w.p. } 1,$$

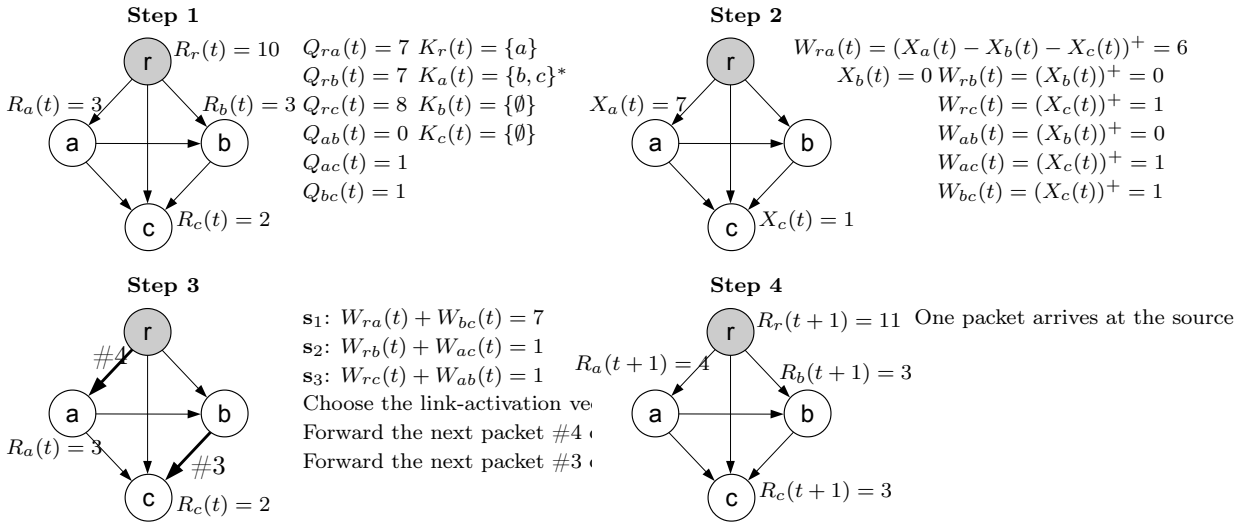


Figure 2-4: Running the optimal broadcast policy π^* in slot t in a wireless network with unit-capacity links and under the primary interference constraint. Step 1: computing the deficits $Q_{ij}(t)$ and $K_j(t)$; a tie is broken in choosing node a as the in-neighbor deficit minimizer for node c, hence $c \in K_a(t)$; node b is also a deficit minimizer for node c. Step 2: computing $X_j(t)$ for $j \neq r$ and $W_{ij}(t)$. Step 3: finding the link activation vector that is a maximizer in (2.17) and forwarding the next in-order packets over the activated links. Step 4: a new packet arrives at the source node r and the values of $\{R_r(t+1), R_a(t+1), R_b(t+1), R_c(t+1)\}$ are updated.

where λ_{DAG} is the upper bound on the broadcast capacity λ^* in the general policy class Π, as given by Eqn. (2.7). Consequently, the broadcast policy π^* achieves the broadcast capacity λ^* in any wireless Directed Acyclic Graph.

Proof See Appendix 2.9.1.

2.3.4 Number of disjoint spanning trees in a DAG

As a corollary, Theorem 2.3.4 yields an interesting combinatorial result that relates the number of disjoint spanning trees in a DAG to the in-degrees of its nodes.

Lemma 2.3.5 *Consider a directed acyclic graph $G = (V, E)$ that is rooted at a node $\mathbf{r}$, has unit-capacity links, and possibly contains parallel edges. The maximum*

number k^* of edge-disjoint spanning trees in G is given by

$$k^* = \min_{v \in V \setminus \{r\}} d_{\text{in}}(v),$$

where $d_{\text{in}}(v)$ denotes the in-degree of the node v.

Proof See Appendix 2.9.2.

2.4 An Efficient Algorithm for Computing the Broadcast Capacity of a Wireless DAG

In this section we exploit Eqn. (2.8) and develop an LP to compute the broadcast capacity of any wireless DAG network under the primary interference constraints. Although this LP has exponentially many constraints, using a well-known separation oracle, it can be solved in strongly polynomial time utilizing the ellipsoid algorithm [37].

Under the primary interference constraint, the set of feasible activations of a graph are its *matchings* [4]. For a subset of edges $E' \subset E$, let $\chi^{E'} \in \{0,1\}^{|E|}$ where $\chi^{E'}(e) = 1$ if $e \in E'$ and is zero otherwise. Let us define

$$\mathcal{P}_{\text{matching}}(\mathcal{G}) = \textbf{convexhull}(\{\chi^M | M \text{ is a matching in } G\}) \tag{2.18}$$

We have the following classical result from Edmonds [38].

Theorem 2.4.1 *The set $\mathcal{P}_{\text{matching}}(\mathcal{G})$ is characterized by the set of all $\boldsymbol{\beta} \in \mathbb{R}^{|E|}$*

such that :

$$\begin{aligned} \beta_e &\geq 0 \quad \forall e \in E \\ \sum_{e \in \delta_{\text{in}}(v) \cup \delta_{\text{out}}(v)} \beta_e &\leq 1 \quad \forall v \in V \\ \sum_{e \in E[U]} \beta_e &\leq \frac{|U|-1}{2}; \quad U \subset V, |U| \text{ odd}, \end{aligned} \tag{2.19}$$

where $E[U]$ is the set of edges (ignoring their directions) with both end points in the subset $U \subset V$.

Hence following Eqn. (2.8), the broadcast capacity of a DAG can be obtained by the following LP :

$$\max \lambda \tag{2.20}$$

Subject to,

$$\lambda \leq \sum_{e \in \delta_{\text{in}}(v)} c_e \beta_e \quad \forall v \in V \setminus \{\mathtt{r}\} \tag{2.21}$$

$$\boldsymbol{\beta} \in \mathcal{P}_{\text{matching}}(\mathcal{G}) \tag{2.22}$$

From the equivalence of optimization and separation (via the ellipsoid method), it follows that the above LP is poly-time solvable if there exists an efficient separator oracle for the constraints (3.16), (3.17). Since there are only linearly many constraints ($|V|-1$, to be precise) in (3.16), the above requirement reduces to an efficient separator for the matching polytope (3.17). We refer to a classic result from the combinatorial-optimization literature which shows the existence of such an efficient separator for the matching polytope.

Theorem 2.4.2 *[38] There exists a strongly poly-time algorithm that, given* $\mathcal{G} = (V, E)$ *and* $\boldsymbol{\beta} : E \to \mathbb{R}^{|E|}$*, determines if* $\boldsymbol{\beta}$ *is feasible for* (3.3.4) *or outputs an inequality from* (3.3.4) *that is violated by* $\boldsymbol{\beta}$*.*

This directly leads to the following theorem.

Theorem 2.4.3 *There exists a strongly poly-time algorithm to compute the broadcast capacity of any wireless DAG under the primary interference constraints.*

The following corollary implies that, although there are exponentially many matchings in a DAG, to achieve the broadcast capacity, randomly activating (with appropriate probabilities) only $|E| + 1$ matchings suffice.

Corollary 2.4.4 *The optimal broadcast capacity* λ^* *in a wireless DAG, under the primary interference constraints, can be achieved by randomly activating (with positive probability) at most* $|E| + 1$ *matchings.*

Proof Let $(\lambda^*, \boldsymbol{\beta^*})$ be an optimal solution of the LP (3.15). Hence we have $\boldsymbol{\beta}^* \in \mathcal{P}_{\text{matching}}(\mathcal{G}) \equiv \textbf{convexhull}(\{\chi^M | M \text{ is a matching in } G\})$. Since the polytope $\mathcal{P}_{\text{matching}}(\mathcal{G})$ is a subset of $\mathbb{R}^{|E|}$, by Carathéodory's theorem [39], the vector $\boldsymbol{\beta}^*$ can be expressed as a convex combination of at most $|E| + 1$ vertices of the polytope $\mathcal{P}_{\text{matching}}(\mathcal{G})$, which are matchings of the graph $\mathcal{G}$. This concludes the proof.

2.5 Broadcasting on Networks with Arbitrary Topology: A Multiclass Algorithm

In this section we extend the above broadcast policy for DAGs to arbitrary networks, which may possibly contain directed cycles. From the negative result of Lemma 2.2.5, we know that any policy ensuring *in-order* packet delivery at every node, cannot achieve the broadcast capacity in arbitrary networks in general. To get around this difficulty, we introduce the notion of broadcasting using multiple *classes* $\mathcal{K}$ of packets. The idea is as follows: each class $k \in \mathcal{K}$ has a one-to-one correspondence with a given permutation $\prec_k$ of the nodes; for an edge $(a, b) \in E$ if the node a appears *before* the node b in the permutation $\prec_k$ (we denote this condition by $a \prec_k b$), then the edge (a, b) is included in the class k, otherwise the edge (a, b) ignored by the class k. The set of all edges included in the class k is denoted by $E^k \subset E$. It is clear that each class k corresponds to a unique embedded DAG topology $\mathcal{G}^k(V, E^k)$, which is a subgraph of the underlying graph $\mathcal{G}(V, E)$. Different classes correspond to different permutations of nodes.

An incoming packet at the source node is admitted to some class $k \in \mathcal{K}$, according to some admission-policy. All packets admitted in a given class $k \in \mathcal{K}$ are broadcasted while maintaining the in-order delivery restriction within the class k, however there is no such inter-class constraint for delivering packets from different classes. Hence the resulting multi-class policy does not belong to the space Π^* but belongs to the general policy-space Π. This new multi-class policy keeps the best from both worlds: (a) its state-space complexity is $\Theta(|\mathcal{K}||V|)$, where for each class we have the same state-representation as in π^* and (b) by relaxing the inter-class in-order delivery constraint, it has the potential to achieve the full broadcast capacity of the underlying graph with sufficiently many classes.

Hence the broadcasting problem reduces to construction of multiple classes (equivalently, permutations of the vertices V) in $\mathcal{G}$ such that they cover the graph *efficiently*, from a broadcast-capacity point of view. In Algorithm-8, we choose the permutations uniformly at random with the condition that the source `r` always appears at the first

position of the permutation.

Algorithm 2 A Multiclass Broadcast Algorithm for Arbitrary Topology

Require: Graph $\mathcal{G}(V, E)$, total number of classes K

1: Generate K permutations $\{\prec_i\}_{i=1}^{K}$ of the nodes V uniformly at random (with the source $\{r\}$ at the first position) and obtain the induced DAGs $G^k(V, E^k)$, where $e = (a, b) \in E^k$ iff $a \prec_k b$.

2: For each permutation $\prec_k$, maintain a class k and the packet-counter variables $\{R_i^{(k)}\}$ at every node $i = 1, 2, \ldots, |V|$.

3: Each class observes intra-class packet forwarding constraints **(1), (2)** and **(3)** described in sections 3.3 and 2.3.

4: Define the state variables $\{\boldsymbol{Q}^k(t), \boldsymbol{X}^k(t)\}$ and compute the weights $\{\boldsymbol{W}^k(t)\}$, for each class $k = 1, 2, \ldots, K$ exactly as in Eqn. (2.16), where each class k considers the edges E^k only for Eqns. (2.15) and (2.16).

5: An incoming packet to source $\mathtt{r}$ at time t joins the class k corresponding to

$$\arg\min_{k \in \mathcal{K}} \sum_{j \in K_{\mathtt{r}}^k(t)} X_j^k(t) \tag{2.23}$$

6: The overall weight for an edge e (taken across all the classes) is computed as

$$W_e(t) = \max_{k: e \in E^k} W_e^k(t) \tag{2.24}$$

7: Activate the edges corresponding to the max-weight activation, i.e.,

$$\boldsymbol{s}(t) \in \arg\max_{\boldsymbol{s} \in \mathcal{S}} \sum_{e \in E} c_e s_e W_e(t). \tag{2.25}$$

8: For each activated edge $e \in \boldsymbol{s}(t)$, forward packets corresponding to a class achieving the maximum in Eqn. (2.24).

Theorem 2.5.1 *The multiclass broadcast Algorithm-8 with K classes supports a broadcast rate of*

$$\lambda^K = \max_{\sum_k \boldsymbol{\beta}^k \in \mathrm{conv}(\mathcal{S})} \sum_{k=1}^{K} \min_{j \neq \boldsymbol{r}} \sum_i c_{ij} \beta_{ij}^k, \tag{2.26}$$

where we use the convention that $\beta_{ij}^k = 0$ if $(i, j) \notin E^{(k)}$.

The right hand side of Eqn. (2.26) can be understood as follows. Consider a feasible stationary activation policy π^{STAT} which activates class k on the edge (i,j) β_{ij}^k fraction of time. Since, by construction, each of the class follows a DAG, lemma (2.3.5) implies that the resulting time-averaged graph has a broadcast capacity of $\lambda^k = \min_j \sum_i c_{ij}\beta_{ij}^k$ for the class k. Thus the total broadcast rate achievable by π^{STAT} is simply $\lambda^K = \sum_{k=1}^K \lambda^k = \sum_k \min_j \sum_i c_{ij}\beta_{ij}^k$. Given these K classes, following the same line of argument as in (3.15), we can develop a similar LP to compute the broadcast-rate achievable (2.26) by these K classes by maximizing over all feasible $\{\boldsymbol{\beta}^k\}_1^K$, in strongly poly-time.

The proof of Theorem (2.5.1) follows along the exact same line of argument as in Theorem (2.3.4), where we now work with the following Lyapunov function $\hat{L}(\boldsymbol{Q}(t))$, which takes into account all K classes:

$$\hat{L}(\boldsymbol{Q}(t)) = \sum_{k=1}^{K} \sum_{j \neq r} (X_j^k(t))^2 \tag{2.27}$$

We then compare the drift of multiclass broadcast algorithm 8 with the stationary randomized policy π^{STAT} above to show that the Multiclass broadcast algorithm is stable under all arrival rates below λ. The details are omitted for brevity.
Since the broadcast-rate λ^K achievable by a collection of K embedded DAGs in a graph $\mathcal{G}$ is always upper-bounded by the actual broadcast capacity λ^* of $\mathcal{G}$, we have the following interesting combinatorial result as a corollary of Theorem (2.5.1)

Corollary 2.5.2 *Consider a wire line network, represented by the graph $\mathcal{G}(V,E)$. For a given integer $K \geq 1$, consider K arbitrary classes (i.e., permutations of vertices) as in Theorem* (2.5.1)*, with $\{E^k\}_{k=1}^K$ being their corresponding edge-sets. Then, for any set of non-negative vectors $\{\boldsymbol{\beta}^k\}_{k=1}^K$ with $\sum_k \beta_{ij}^k \leq 1, \forall (i,j)$, the following lower-bound for the broadcast capacity λ^* holds:*

$$\lambda^* \geq \sum_{k=1}^{K} \min_{j \neq r} \sum_i c_{ij}\beta_{ij}^k \tag{2.28}$$

where we use the convention that $\beta_{ij}^k = 0$ if $(i,j) \notin E^k$.

The above corollary may be contrasted with Eqn. (2.7), which provides an upper bound to the broadcast capacity λ^*. We also note that, the lower-bound in Eqn. (2.28) is tight when the classes are chosen corresponding to the maximum number of edge-disjoint spanning trees, obtained from Edmonds' Theorem [14].

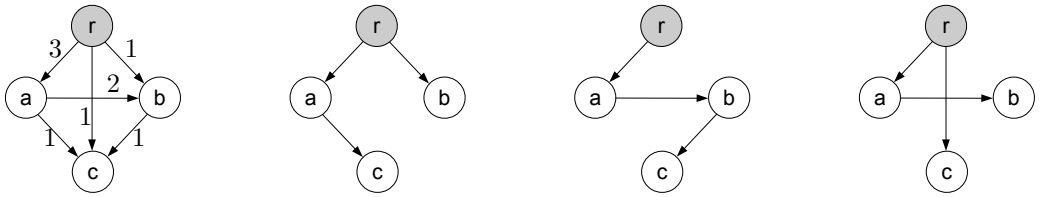
2.6 Simulation Results

We present a number of simulation results concerning the delay performance of the optimal broadcast policy π^* in wireless DAG networks with different topologies. For simplicity, we assume primary interference constraints for wireless networks throughout this section. Delay for a packet is defined as the number of slots required for it to reach *all* nodes in the network, after its arrival to the source `r`.

Diamond topology

Consider a 4-node wireless network as shown Fig. 2-5 (a). Link capacities are indicated alongside the links. The broadcast capacity of the network is upper bounded by the total capacity of incoming links to node `c`, which is 1. This is because at most one of its unit-capacity incoming links to node `c` may be activated at any slot, under the primary interference constraint. To determine the broadcast-capacity of the network, consider three spanning trees $\{\mathcal{T}_1, \mathcal{T}_2, \mathcal{T}_3\}$ rooted at the source node `r`, as shown in Fig. 2-5 (b),(c),(d). By finding an optimal time-sharing of all feasible link-activations over a subset of spanning trees using linear programming and using Eqn. (2.26), we can show that the broadcast-rate achievable using the tree $\mathcal{T}_1$ only is 3/4, using the trees $\{\mathcal{T}_1, \mathcal{T}_2\}$ only is 6/7, and using the trees $\{\mathcal{T}_1, \mathcal{T}_2, \mathcal{T}_3\}$ together is 1. Thus, the upper-bound is achieved and the broadcast capacity of the network is $\lambda^* = 1$.

We compare the performance of our throughput-optimal broadcast policy π^* with the tree-based policy π_{tree} proposed in [2]. While the policy π_{tree} is originally proposed to transmit multicast traffic in a wired network by balancing traffic over multiple trees, we generalize their policy π_{tree} for broadcasting packets over spanning trees in the wireless setting. Fig. 2.3.3 shows a comparison of the average delay performance



(a) The wireless network (b) Tree $\mathcal{T}_1$ (c) Tree $\mathcal{T}_2$ (d) Tree $\mathcal{T}_3$

Figure 2-5: A wireless DAG network and its three embedded spanning trees.

under the policy π^* and the tree-based policy π_{tree} over different subset of trees. The simulation duration is 10^5 slots. We observe that the policy π^* achieves the broadcast capacity and, in general, has better delay performance than the tree-based scheme in the high traffic regime.

Mesh topology

Since the throughput-optimal broadcast policy π^* does not rely on limited number of tree structures, it has the potential to exploit all degrees of freedom in the network. Such freedom leads to better delay performance as compared to the tree-based broadcast policies [2]. To illustrate this effect, consider the 10-node DAG network in Fig. 2-7 (a). For every pair of node $\{i, j\}$, $1 \leq i < j \leq 10$, the network has a directed link from node i to j with capacity $(10 - i)$. By induction, the number of spanning trees rooted at the source node 1 can be calculated to be $9! \approx 3.6 \times 10^5$. Among them, we choose five arbitrary spanning trees $\{\mathcal{T}_i, 1 \leq i \leq 5\}$, shown in Fig. 2-7 (b),(c),(d),(e),(f), over which the tree-based algorithm π_{tree} is simulated. Table 2.1 demonstrates the superior delay performance of our throughput-optimal broadcast policy π^*, as compared to that of the tree-based algorithm π_{tree}. The table also shows that a tree-based algorithm that does not use enough number of trees might result in degraded broadcast throughput.

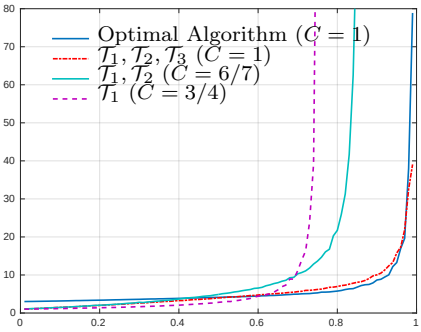
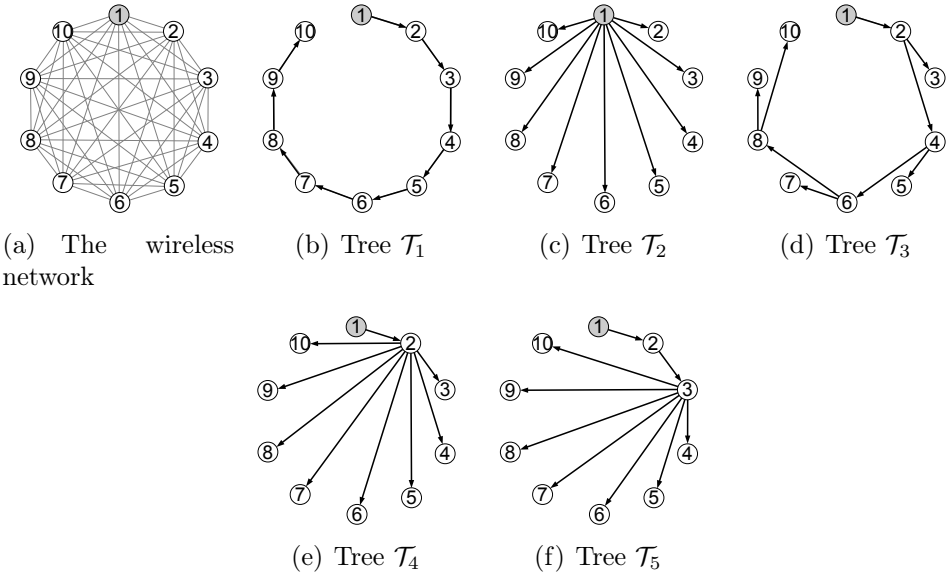


Figure 2-6: Average delay performance of the optimal broadcast policy π^* and the tree-based policy π_{tree} that balances traffic over different subsets of spanning trees.



(a) The wireless network (b) Tree $\mathcal{T}_1$ (c) Tree $\mathcal{T}_2$ (d) Tree $\mathcal{T}_3$

(e) Tree $\mathcal{T}_4$ (f) Tree $\mathcal{T}_5$

Figure 2-7: The 10-node wireless DAG network and a subset of spanning trees.

Multiclass Simulation for Arbitrary Topology

To simulate the multiclass broadcast algorithm of section 2.5, we randomly generate an ensemble of 500 wire line networks (not necessarily DAGs), each consisting of $N = 10$ nodes and unit capacity links. By solving the LP corresponding to Eqn. (2.26), we compute the average fraction of the total broadcast capacity achievable using K randomly chosen classes by the Multiclass Algorithm 8 of section 2.5. The result is plotted in Figure 2-8. It follows that a sizeable fraction of the optimal capacity may be achieved by using a moderately many classes. However, it also shows that the required number of classes for achieving a certain fraction of the capacity increases as

λ	Tree-based policy π_{tree} over the spanning trees					Broadcast policy π^*
	$\mathcal{T}_1$	$\mathcal{T}_1 \sim \mathcal{T}_2$	$\mathcal{T}_1 \sim \mathcal{T}_3$	$\mathcal{T}_1 \sim \mathcal{T}_4$	$\mathcal{T}_1 \sim \mathcal{T}_5$	
0.5	12.90	12.72	13.53	16.14	16.2	11.90
0.9	1.3×10^4	176.65	106.67	34.33	28.31	12.93
1.9	3.31×10^4	1.12×10^4	4.92×10^3	171.56	95.76	14.67
2.3	3.63×10^4	1.89×10^4	1.40×10^4	1.76×10^3	143.68	17.35
2.7	3.87×10^4	2.45×10^4	2.03×10^4	1.1×10^4	1551.3	20.08
3.1	4.03×10^4	2.86×10^4	2.51×10^4	1.78×10^4	9788.1	50.39

Table 2.1: Average delay performance of the tree-based policy π_{tree} over different subsets of spanning trees and the broadcast policy π^*.

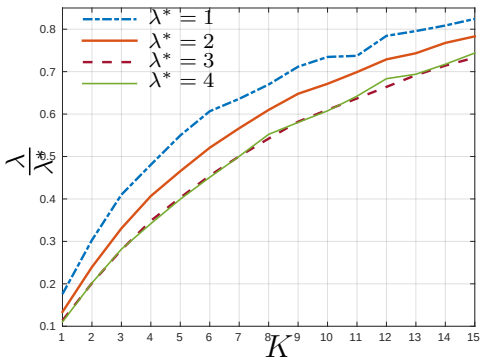


Figure 2-8: Fraction of optimal broadcast rate $\frac{\lambda}{\lambda^*}$ achievable by the multiclass broadcast algorithm with randomly chosen K classes for randomly generated wired networks with $N = 10$ nodes.

the broadcast capacity of the network increases. This is due to the fact that increased broadcast capacity of a network would warrant an increased number of DAGs to cover it efficiently.

2.7 Conclusion

In this chapter we initiated our study of the problem of broadcasting in a static wireless network under general interference constraints. When the underlying network topology is a DAG, we proposed a dynamic algorithm that achieves the broadcast capacity of the network. Our novel algorithm, based on packet deficits and the in-order packet delivery constraint, is promising for application to other systems with packet duplications, such as multicasting and caching systems. We also propose a heuristic extension of our DAG broadcast algorithm to networks with arbitrary

topology. In the next chapter we will extend some of the methodologies introduced in this chapter to design a throughput-optimal broadcast policy for time-varying wirless DAG networks.

2.8 Appendix

2.8.1 Proof of Theorem 2.2.3

Fix an $\epsilon > 0$. Consider a broadcast policy $\pi \in \Pi$ that achieves a broadcast rate of at least $\lambda^* - \epsilon$, as defined in (4.1); this policy π exists by the definition of the broadcast capacity λ^* in Definition 4.2.2. Consider any proper cut U of the network $\mathcal{G}$. By definition of a proper-cut, there exists a node $i \notin U$. Let $\boldsymbol{s}^\pi(t) = (s_e^\pi(t), e \in E)$ be the link-activation vector chosen by policy π in slot t. The maximum number of packets that can be transmitted across the cut U by any policy in slot t is at most $\sum_{e \in E_U} c_e s_e^\pi(t)$, which is the total capacity of all activated links across the outgoing-edges from the cut U, where the link subset E_U has been defined in Eqn. (2.3). Thus, the number of distinct packets $R_i^\pi(T)$ received by a node i by time T can be upper-bounded as follows

$$R_i^\pi(T) \leq \sum_{t=1}^{T} \sum_{e \in E_U} c_e s_e^\pi(t) = \boldsymbol{u} \cdot \sum_{t=1}^{T} \boldsymbol{s}^\pi(t), \tag{2.29}$$

where we define the $|E|$-dimensional cut-vector $\boldsymbol{u} = (u_e, e \in E)$, such that $u_e = c_e 1_{[e \in E_U]}$, and $\boldsymbol{a} \cdot \boldsymbol{b}$ is the inner product of two vectors.[5] Dividing both sides by T yields

$$\frac{R_i^\pi(T)}{T} \leq \boldsymbol{u} \cdot \left(\frac{1}{T} \sum_{t=1}^{T} \boldsymbol{s}^\pi(t) \right).$$

[5]Note that Eqn. (3.6) remains valid even if the network coding operations are allowed.

Hence,

$$\lambda^* - \epsilon \overset{(a)}{\leq} \min_{j \in V} \liminf_{T \to \infty} \frac{R_j^\pi(T)}{T} \leq \liminf_{T \to \infty} \frac{R_i^\pi(T)}{T}$$
$$\leq \liminf_{T \to \infty} \boldsymbol{u} \cdot \left(\frac{1}{T} \sum_{t=1}^{T} \boldsymbol{s}^\pi(t) \right), \tag{2.30}$$

where (a) follows because π is assumed to be a broadcast policy of rate at least $\lambda^* - \epsilon$. Since the above holds for any proper-cut $\boldsymbol{u} \in \mathcal{U}$, we have

$$\lambda^* - \epsilon \leq \min_{\boldsymbol{u} \in \mathcal{U}} \liminf_{T \to \infty} \boldsymbol{u} \cdot \left(\frac{1}{T} \sum_{t=1}^{T} \boldsymbol{s}^\pi(t) \right) \tag{2.31}$$

Now consider the following lemma.

Lemma 2.8.1 *For any policy $\pi \in \Pi$, there exists a vector $\boldsymbol{\beta}^\pi \in \text{conv}(\mathcal{S})$ such that*

$$\min_{\boldsymbol{u} \in \mathcal{U}} \liminf_{T \to \infty} \boldsymbol{u} \cdot \left(\frac{1}{T} \sum_{t=1}^{T} \boldsymbol{s}^\pi(t) \right) = \min_{\boldsymbol{u} \in \mathcal{U}} \boldsymbol{u} \cdot \boldsymbol{\beta}^\pi$$

Proof Consider a sequence of vectors $\boldsymbol{\zeta}_T^\pi \overset{\text{def}}{=} \frac{1}{T} \sum_{t=1}^{T} \boldsymbol{s}^\pi(t)$, indexed by $T \geq 1$. Since $\boldsymbol{s}^\pi(t) \in \mathcal{S}$ for all $t \geq 1$, we have $\boldsymbol{\zeta}_T^\pi \in \text{conv}\,(\mathcal{S})$ for all $T \geq 1$. Since $|\mathcal{U}|$ is finite, by the definition of lim inf, there exists a sub-sequence $\{\boldsymbol{u} \cdot \boldsymbol{\zeta}_{T_k}^\pi\}_{k \geq 1}$ of the sequence $\{\boldsymbol{u} \cdot \boldsymbol{\zeta}_T^\pi\}_{T \geq 1}$ such that

$$\min_{\boldsymbol{u} \in \mathcal{U}} \lim_{k \to \infty} \boldsymbol{u} \cdot \boldsymbol{\zeta}_{T_k}^\pi = \min_{\boldsymbol{u} \in \mathcal{U}} \liminf_{T \to \infty} \boldsymbol{u} \cdot \boldsymbol{\zeta}_T^\pi. \tag{2.32}$$

Since the set $\text{conv}\,(\mathcal{S}) \subset \mathbb{R}^{|E|}$ is closed and bounded, by the Heine-Borel theorem, it is compact. Hence any sequence in $\text{conv}(\mathcal{S})$ has a converging sub-sequence. Thus, there exists a sub-sub-sequence $\{\boldsymbol{\zeta}_{T_{k_i}}^\pi\}_{i \geq 1}$ and $\boldsymbol{\beta}^\pi \in \text{conv}(\mathcal{S})$ such that

$$\boldsymbol{\zeta}_{T_{k_i}}^\pi \to \boldsymbol{\beta}^\pi, \quad \text{as } i \to \infty.$$

It follows that

$$\begin{aligned}
\min_{\boldsymbol{u}\in\mathcal{U}} \boldsymbol{u}\cdot\boldsymbol{\beta}^{\pi} &\overset{(a)}{=} \min_{\boldsymbol{u}\in\mathcal{U}} \lim_{i\to\infty} \boldsymbol{u}\cdot\boldsymbol{\zeta}^{\pi}_{T_{k_i}} \\
&\overset{(b)}{=} \min_{\boldsymbol{u}\in\mathcal{U}} \lim_{k\to\infty} \boldsymbol{u}\cdot\boldsymbol{\zeta}^{\pi}_{T_{k}} \\
&\overset{(c)}{=} \min_{\boldsymbol{u}\in\mathcal{U}} \liminf_{T\to\infty} \boldsymbol{u}\cdot\boldsymbol{\zeta}^{\pi}_{T} \\
&\overset{(d)}{=} \min_{\boldsymbol{u}\in\mathcal{U}} \liminf_{T\to\infty} \boldsymbol{u}\cdot\left(\frac{1}{T}\sum_{t=1}^{T}\boldsymbol{s}^{\pi}(t)\right),
\end{aligned}$$

where (a) uses the fact that if $\boldsymbol{x}_n \to \boldsymbol{x}$ then $\boldsymbol{c}\cdot\boldsymbol{x}_n \to \boldsymbol{c}\cdot\boldsymbol{x}$ for any $\boldsymbol{c}$, $\boldsymbol{x}_n$, and $\boldsymbol{x} \in \mathbb{R}^l$, $l \geq 1$; (b) follows from the fact that if the limit of a sequence $\{z_k \equiv \boldsymbol{u}\cdot\boldsymbol{\zeta}^{\pi}_{T_k}\}$ exists then all sub-sequences $\{z_{k_i} \equiv \boldsymbol{u}\cdot\boldsymbol{\zeta}^{\pi}_{T_{k_i}}\}$ converge and $\lim_i z_{k_i} = \lim_k z_k$; (c) follows from Equation (3.38) and (d) follows from the definition of the sequence $\boldsymbol{\zeta}^{\pi}_{T}$. This completes the proof of the lemma.

Combining Lemma 3.3.1 with Eqn. (3.8), we have that there exists a vector $\boldsymbol{\beta}^{\pi} \in \text{conv}(\mathcal{S})$ such that

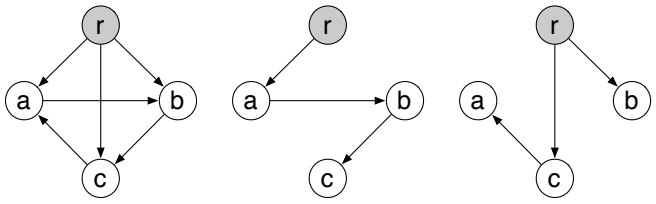
$$\lambda^* - \epsilon \leq \min_{\boldsymbol{u}\in\mathcal{U}} \boldsymbol{u}\cdot\boldsymbol{\beta}^{\pi}. \tag{2.33}$$

Maximizing the right hand side of Eqn. 3.9 over all $\boldsymbol{\beta}^{\pi} \in \text{conv}(\mathcal{S})$, we have

$$\lambda^* - \epsilon \leq \max_{\boldsymbol{\beta}\in\text{conv}(\mathcal{S})} \left(\min_{\boldsymbol{u}\in\mathcal{U}} \boldsymbol{u}\cdot\boldsymbol{\beta}\right) \tag{2.34}$$

Since the above inequality holds for any $\epsilon > 0$, by taking $\epsilon \searrow 0$ and expanding the dot product, we have

$$\lambda^* \leq \max_{\boldsymbol{\beta}\in\text{conv}(\mathcal{S})} \left(\min_{U:\text{a proper cut}} \sum_{e\in E_U} c_e\beta_e\right). \tag{2.35}$$



(a) A wired network with a directed cycle $a \to b \to c \to a$.

(b) Tree $\mathcal{T}_1$

(c) Tree $\mathcal{T}_2$

Figure 2-9: A wired network and its two edge-disjoint spanning trees that yield the broadcast capacity $\lambda^* = 2$.

2.8.2 Proof of Lemma 2.2.5

2.9 Proof of Lemma 1

Consider the non-DAG wire line network of Fig. 2-9(a), where all edges have unit capacity and there is no interference constraint. Since the sum of edge-capacities of the links incoming to node a is 2, its throughput, and hence the broadcast capacity of the network is upper bounded by 2. In fact, the network has two edge-disjoint directed spanning trees rooted at the source `r`, as shown in Figures 2-9(b) and 2-9(c). Hence, we can achieve the broadcast capacity $\lambda^* = 2$, e.g., by broadcasting the odd-numbered and even-numbered packets along the trees $\mathcal{T}_1$ and $\mathcal{T}_2$, respectively.

Consider a broadcast policy $\pi \in \Pi^{\text{in-order}}$ that ensures in-order delivery of packets to all nodes. Let $R_i(t)$ be the number of distinct packets received by node i up to time t. Hence, the node i has received the set of packets $\{1, 2, \ldots, R_i(t)\}$ by time t, due to the property of in-order delivery. Consider the directed cycle $a \to b \to c \to a$ in Fig. 2-9(a). A necessary condition for all links in the cycle to forward (non-duplicate) packets in slot t is $R_a(t) > R_b(t) > R_c(t) > R_a(t)$, which is clearly impossible. Thus, there must exist an idle link in the cycle at every slot. Define the indicator variable $x_e(t) = 1$ if link e is idle in slot t under the policy π, and $x_e(t) = 0$ otherwise. Since at least one link in the cycle is idle in every slot, we have

$$x_{(a,b)}(t) + x_{(b,c)}(t) + x_{(c,a)}(t) \geq 1.$$

Taking the time average of the above inequality yields

$$\frac{1}{T}\sum_{t=1}^{T}\big(x_{(a,b)}(t)+x_{(b,c)}(t)+x_{(c,a)}(t)\big)\geq 1.$$

Taking the lim sup at both sides, we obtain

$$\sum_{e\in\{(a,b),(b,c),(c,a)\}}\limsup_{T\to\infty}\frac{1}{T}\sum_{t=1}^{T}x_e(t) \quad\geq\quad \limsup_{T\to\infty}\sum_{e\in\{(a,b),(b,c),(c,a)\}}\frac{1}{T}\sum_{t=1}^{T}x_e(t) \quad\geq\quad 1.$$

The above inequality implies that

$$\max_{e\in\{(a,b),(b,c),(c,a)\}}\limsup_{T\to\infty}\frac{1}{T}\sum_{t=1}^{T}x_e(t)\geq\frac{1}{3}. \tag{2.36}$$

Since the nodes $\{a,b,c\}$ are symmetrically located (i.e., the graph obtained by permuting the nodes $\{a,b,c\}$ is isomorphic to the original graph), without any loss of generality, we may assume that the link $e=(a,b)$ attains the maximum in the LHS of the inequality (2.36), i.e.,

$$\limsup_{T\to\infty}\frac{1}{T}\sum_{t=1}^{T}x_{(a,b)}(t)\geq\frac{1}{3}. \tag{2.37}$$

Noting that $x_e(t)=1$ if link e is idle in slot t and that node b receives packets only from nodes $\mathtt{r}$ and a, we can upper bound $R_b(T)$ by

$$R_b(T)\leq\sum_{t=1}^{T}\big(1-x_{(r,b)}(t)+1-x_{(a,b)}(t)\big)\leq\sum_{t=1}^{T}\big(2-x_{(a,b)}(t)\big).$$

From the above it follows that,

$$\liminf_{T\to\infty}\frac{R_b(T)}{T}\leq 2-\limsup_{T\to\infty}\frac{1}{T}\sum_{t=1}^{T}x_{(a,b)}(t)\leq\frac{5}{3},$$

where the last inequality uses (2.37). Thus, we have

$$\min_{i\in V} \liminf_{T\to\infty} \frac{R_i(T)}{T} \leq \liminf_{T\to\infty} \frac{R_b(T)}{T} \leq \frac{5}{3},$$

which holds for all policies $\pi \in \Pi^{\text{in-order}}$. Taking supremum over the policy class $\Pi^{\text{in-order}}$ shows that the broadcast capacity $\lambda^*_{\text{in-order}}$ subject to the in-order delivery constraint satisfies

$$\lambda^*_{\text{in-order}} = \sup_{\pi\in\Pi^{\text{in-order}}} \min_{i\in V} \liminf_{T\to\infty} \frac{R_i(T)}{T} \leq \frac{5}{3} < 2 = \lambda^*.$$

i.e., the broadcast capacity of the network is strictly reduced by the in-order packet delivery constraint in the non-DAG network of Fig. 2-9(a).

2.9.1 Proof of Theorem 2.3.4

We complete the proof in four steps. First, using the dynamics of $X_j(t)$ in the space Π^* (Eqn. (2.14)), we derive an expression of one-slot drift of an appropriately defined Quadratic Lyapunov function $L(\boldsymbol{X}(t))$. Second, we design an auxiliary stationary randomized policy π^{RAND} for link-activations that yields optimal broadcast throughput. Third, this randomized policy is used to show that the system $\boldsymbol{X}(t)$ is *strongly stable* for all arrival rates $\lambda < \lambda^*$, under the optimal broadcast policy $\pi^* \in \Pi^*$. Finally, based on the above analysis, we finally show that the policy π^* is a throughput-optimal broadcast policy for any wireless DAG network.

(a) An Upper-bound on the drift of the policy π^* :

Lemma 2.9.1 *For the dynamics*

$$Q(t+1) \leq (Q(t) - \mu(t))^+ + A(t) \tag{2.38}$$

where all variables are non-negative and $(x)^+ \stackrel{def}{=} \max\{x, 0\}$, we have,

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

Proof Squaring both sides of Eqn. (3.41), we have

$$\begin{aligned} Q^2(t+1) &\le \big((Q(t)-\mu(t))^+\big)^2 + A^2(t) + 2A(t)(Q(t)-\mu(t))^+ \\ &\le (Q(t)-\mu(t))^2 + A^2(t) + 2A(t)Q(t), \end{aligned}$$

where we use the fact that $x^2 \ge (x^+)^2$, $Q(t) \ge 0$, and $\mu(t) \ge 0$. Rearranging the above inequality finishes the proof.

Applying Lemma 3.8.1 to the dynamics (2.14) of $X_j(t)$ yields, for each node $j \ne \mathtt{r}$,

$$X_j^2(t+1) - X_j^2(t) \le Y(t) + 2X_j(t)\big(\sum_{m\in V}\mu_{mi_t^*}(t) - \sum_{k\in V}\mu_{kj}(t)\big), \tag{2.39}$$

where $Y(t) \stackrel{\text{def}}{=} (\sum_{m\in V}\mu_{mi_t^*}(t))^2 + (\sum_{k\in V}\mu_{kj}(t))^2$.

Let $C = \sum_e c_e$, the sum of the capacities of all links in the network. Now the node i_t^* could be the source node $\mathtt{r}$ or a non-source node in the network. In either case, since $\mu_e(t) \le c_e, \forall e \in E$, the first term in $Y(t)$ above is upper-bounded by $\max\{A^2(t), C^2\}$ and the second term is upper-bounded by C^2. Hence, $Y(t) \le \max\{A^2(t), C^2\} + C^2 \le A^2(t) + 2C^2$. Since the number of arrivals per slot $A(t)$ is assumed to have bounded second moment, there exists a finite constant $B > 0$ such that $\mathbb{E}[Y(t)] \le \mathbb{E}\big(A^2(t)\big) + 2C^2 \le B$.

Now define a Quadratic Lyapunov function $L(\boldsymbol{X}(t)) \stackrel{\text{def}}{=} \sum_{j\ne\mathtt{r}} X_j^2(t)$. From Eqn. (3.42), the one-slot drift $\Delta(\boldsymbol{X}(t))$ of $L(\boldsymbol{X}(t))$ may be computed to be

$$\begin{aligned} &\Delta(\boldsymbol{X}(t)) \triangleq \mathbb{E}[L(\boldsymbol{X}(t+1)) - L(\boldsymbol{X}(t)) \mid \boldsymbol{X}(t)] \\ &= \mathbb{E}\big[\sum_{j\ne\mathtt{r}}\big(X_j^2(t+1) - X_j^2(t)\big) \mid \boldsymbol{X}(t)\big] \\ &\le B|V| + 2\sum_{j\ne\mathtt{r}} X_j(t)\mathbb{E}\big[\sum_{m\in V}\mu_{mi_t^*}(t) - \sum_{k\in V}\mu_{kj}(t) \mid \boldsymbol{X}(t)\big] \\ &\stackrel{(a)}{=} B|V| - 2\sum_{(i,j)\in E}\mathbb{E}[\mu_{ij}(t) \mid \boldsymbol{X}(t)]\big(X_j(t) - \sum_{k\in K_j(t)} X_k(t)\big) \\ &= B|V| - 2\sum_{(i,j)\in E}\mathbb{E}[\mu_{ij}(t) \mid \boldsymbol{X}(t)]\, W_{ij}(t), \end{aligned} \tag{2.40}$$

where (a) follows from changing the order of summation and $K_j(t)$ and $W_{ij}(t)$ are as defined in Eqn. (2.15) and (2.16), respectively. To emphasize the fact that the drift upper-bound (3.44) depends on the control policy $\pi \in \Pi^*$, we attach a superscript π to the control variables $\boldsymbol{\mu}(t)$ as follows:

$$\Delta^{\pi}(\boldsymbol{X}(t)) \leq B|V| - 2 \sum_{(i,j)\in E} \mathbb{E}[\mu_{ij}^{\pi}(t) \mid \boldsymbol{X}(t)]\, W_{ij}(t). \tag{2.41}$$

Our optimal broadcast policy $\pi^* \in \Pi^*$ is chosen to minimize the upper-bound on the drift expression, given by the right-hand side of Eqn. (2.41), among all policies in the space Π^*.

(b) Construction of a Stationary Randomized Policy π^{RAND} : Next, we construct an auxiliary randomized link-activation policy π^{RAND}, which will be useful later in the proof. Let the vector $\boldsymbol{\beta}^* \in \mathrm{conv}\,(\mathcal{S})$ attain the upper-bound in Eqn. (2.5):

$$\boldsymbol{\beta}^* \in \arg \max_{\boldsymbol{\beta}\in \mathrm{conv}(\mathcal{S})} \min_{U\text{: a proper cut}} \sum_{e\in E_U} c_e \beta_e.$$

From Caratheodory's theorem [39], there exist at most $(|E|+1)$ link-activation vectors $\{\boldsymbol{s}_k \in \mathcal{S}\}$ and associated non-negative scalars $\{p_k \geq 0\}$ with $\sum_{k=1}^{|E|+1} p_k = 1$, such that

$$\boldsymbol{\beta}^* = \sum_{k=1}^{|E|+1} p_k \boldsymbol{s}_k. \tag{2.42}$$

Hence, from Theorem 2.2.3 we have,

$$\lambda^* \leq \min_{U\text{: a proper cut}} \sum_{e\in E_U} c_e \beta_e^*. \tag{2.43}$$

Consider an exogenous packet arrival rate λ at the source, which is strictly less than the broadcast capacity λ^*. Thus, there exists an $\epsilon > 0$ such that $\lambda + \epsilon \leq \lambda^*$. From Eqn. (3.49),

$$\lambda + \epsilon \leq \min_{U\text{: a proper cut}} \sum_{e\in E_U} c_e \beta_e^*. \tag{2.44}$$

For any node $v \neq \mathtt{r}$ other than the source, consider the specific proper cuts $U_v =$

$V \setminus \{v\}$, defined earlier in Eqn. (2.6). From Eqn. (3.50), we have

$$\lambda + \epsilon \leq \sum_{e \in E_{U_v}} c_e \beta_e^*, \ \forall v \neq \mathtt{r}. \tag{2.45}$$

Since the underlying network topology $\mathcal{G} = (V, E)$ is a DAG, there exists a topological ordering of the nodes such that: (i) the nodes can be labelled serially as $\{v_1, \ldots, v_{|V|}\}$, where $v_1 = \mathtt{r}$ is the source node with no in-neighbours and the node $v_{|V|}$ has no outgoing neighbours and (ii) all edges in E are oriented from $v_i \to v_j$, $i < j$ [40]; From Eqn. (3.51), we define probabilities $q_j \in [0, 1]$ for each node v_j such that

$$q_j \sum_{e \in E_{U_{v_j}}} c_e \beta_e^* = \lambda + \epsilon \frac{j}{|V|}, \ j = 2, \ldots, |V|. \tag{2.46}$$

Consider a randomized link-activation policy π^{RAND} defined as follows: at every slot t (i) it randomly *selects* a feasible link-activation vector $\boldsymbol{s}(t) = \boldsymbol{s}_k$ with probability p_k, given in Eqn. (3.47), $k = 1, 2, \ldots, |E| + 1$; (ii) for each selected link $e = (v_i, v_j)$, incoming to the node v_j with $s_e(t) = 1$, the link e is activated independently with probability q_j, given by Eqn. (3.52). The activated links are used to forward packets, subject to the constraints that define the policy class Π^* (i.e., in-order packet delivery and that a network node is only allowed to receive packets that have been received by all of its in-neighbors). Note that this randomized policy is independent of the state $\boldsymbol{X}(t)$. Since each node $j \in V$ is relabeled by the topological ordering as $v_l \in V$ for some $2 \leq l \leq |V|$, from Eqn. (3.52) we conclude that, for each node $j \neq \mathtt{r}$, the total expected incoming transmission rate to node j is given by

$$\begin{aligned} &\sum_{i:(i,j) \in E} \mathbb{E}[\mu_{ij}^{\pi^{\text{RAND}}}(t) \mid \boldsymbol{X}(t)] = \sum_{i:(i,j) \in E} \mathbb{E}[\mu_{ij}^{\pi^{\text{RAND}}}(t)] \\ &= q_l \sum_{e \in E_{U_{v_l}}} c_e \beta_e^* = \lambda + \epsilon \frac{l}{|V|}. \end{aligned} \tag{2.47}$$

Equation (3.53) shows that under the randomized policy π^{RAND}, the total expected incoming capacity to each node $j \neq \mathtt{r}$ is strictly larger than the packet arrival rate λ.

According to the abuse of notation in (2.14), at the source node $\mathtt{r}$ we have

$$\sum_{i:(i,\mathbf{r})\in E} \mathbb{E}[\mu_{i\mathbf{r}}^{\pi^{\text{RAND}}}(t) \mid \boldsymbol{X}(t)] = \mathbb{E}[\sum_{i:(i,\mathbf{r})\in E} \mu_{i\mathbf{r}}^{\pi^{\text{RAND}}}(t)] = \lambda. \tag{2.48}$$

From Eqns. (3.53) and (3.54), if node i appears prior to node j in the aforementioned topological ordering, i.e., if $i \equiv v_{l_i} < v_{l_j} \equiv j$ for some $l_i < l_j$, then

$$\sum_{k:(k,i)\in E} \mathbb{E}[\mu_{ki}^{\pi^{\text{RAND}}}(t) \mid \boldsymbol{X}(t)] - \sum_{k:(k,j)\in E} \mathbb{E}[\mu_{kj}^{\pi^{\text{RAND}}}(t) \mid \boldsymbol{X}(t)] \leq -\frac{\epsilon}{|V|} \tag{2.49}$$

(c) Stochastic Stability of $\{\boldsymbol{X}(t)\}_{t\geq 0}$ under π^* : The drift inequality (2.41) holds for any policy $\pi \in \Pi^*$. Our broadcast policy π^* observes the system state $\boldsymbol{X}(t)$ and minimizes the upper-bound on drift at every slot. Comparing the activations selected by the policy π^* with π^{RAND} in slot t, we have

$$\begin{aligned}
\Delta^{\pi^*}(\boldsymbol{X}(t)) &\leq B|V| - 2\sum_{(i,j)\in E} \mathbb{E}\big[\mu_{ij}^{\pi^*}(t) \mid \boldsymbol{X}(t)\big] W_{ij}(t) \\
&\leq B|V| - 2\sum_{(i,j)\in E} \mathbb{E}\big[\mu_{ij}^{\pi^{\text{RAND}}}(t) \mid \boldsymbol{X}(t)\big] W_{ij}(t) \\
&= B|V| + 2\sum_{j\neq \mathbf{r}} X_j(t)\Bigg(\sum_{m\in V} \mathbb{E}\big[\mu_{mi_t^*}^{\pi^{\text{RAND}}}(t) \mid \boldsymbol{X}(t)\big] - \sum_{k\in V} \mathbb{E}\big[\mu_{kj}^{\pi^{\text{RAND}}}(t) | \boldsymbol{X}(t)\big]\Bigg) \\
&\leq B|V| - \frac{2\epsilon}{|V|}\sum_{j\neq r} X_j(t).
\end{aligned} \tag{2.50}$$

Since node i_t^* is an in-neighbour of node j (2.10), the node i_t^* must appear before j in any topological ordering of the DAG $\mathcal{G}$. Hence, the inequality in (3.59) follows directly from (3.55). Taking expectation of both sides in (3.59) with respect to $\boldsymbol{X}(t)$,

$$\mathbb{E}\big[L(\boldsymbol{X}(t+1))\big] - \mathbb{E}\big[L(\boldsymbol{X}(t))\big] \leq B|V| - \frac{2\epsilon}{|V|}\mathbb{E}||\boldsymbol{X}(t)||_1,$$

where $||\cdot||_1$ is the ℓ_1-norm. Summing the above inequality over $t = 0, 1, 2, \ldots T-1$ yields

$$\mathbb{E}\big[L(\boldsymbol{X}(T))\big] - \mathbb{E}\big[L(\boldsymbol{X}(0))\big] \leq B|V|T - \frac{2\epsilon}{|V|}\sum_{t=0}^{T-1} \mathbb{E}||\boldsymbol{X}(t)||_1.$$

Dividing the above by $2T\epsilon/|V|$ and using $L(\boldsymbol{X}(T)) \geq 0$,

$$\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}||\boldsymbol{X}(t)||_1 \leq \frac{B|V|^2}{2\epsilon} + \frac{|V|\,\mathbb{E}[L(\boldsymbol{X}(0))]}{2T\epsilon}$$

Taking a lim sup of both sides as $T \to \infty$, we have

$$\limsup_{T\to\infty}\frac{1}{T}\sum_{t=0}^{T-1}\sum_{j\neq \mathtt{r}}\mathbb{E}[X_j(t)] \leq \frac{B|V|^2}{2\epsilon}, \tag{2.51}$$

which implies that the virtual-queue process $\{\boldsymbol{X}(t)\}_{t=0}^{\infty}$ is strongly stable [33] under the policy $\pi^* \in \Pi^*$.

(d) Throughput-optimality of π^* : Finally, we show that the strong stability of the virtual queues $X_j(t)$ implies that the policy π^* achieves the broadcast capacity λ^* in a DAG, i.e., for all arrival rates $\lambda < \lambda^*$, we have

$$\lim_{T\to\infty}\frac{R_j(T)}{T} = \lambda,\ \forall j.$$

Equation (2.14) shows that the virtual queues $X_j(t)$ have bounded departures (due to the bounded link capacities). Thus, strong stability of $X_j(t)$ implies that all virtual queues $X_j(t)$ are rate stable [33, Theorem 2.8], i.e., $\lim_{T\to\infty} X_j(T)/T = 0$, w.p.1 for all j. Using union-bound, it follows that,

$$\lim_{T\to\infty}\frac{1}{T}\sum_{j\neq \mathtt{r}} X_j(T) = 0, \quad \text{w.p. 1} \tag{2.52}$$

Now consider any node $j \neq \mathtt{r}$ in the network. We can construct a simple path $\sigma(\mathtt{r} \stackrel{\text{def}}{=} u_k \to u_{k-1} \ldots \to u_1 \stackrel{\text{def}}{=} j)$ from the source node $\mathtt{r}$ to the node j by running Algorithm 3 on the DAG $\mathcal{G}(V, E)$.

Algorithm 3 chooses the parent of a node u_i in the path σ as the one that has the least relative packet deficit as compared to u_i. Since the underlying graph $\mathcal{G}(V, E)$ is a connected DAG (i.e., there is a path from the source to every other node in the network), the above path construction algorithm always terminates with a path

Algorithm 3 $\mathtt{r} \to j$ Path Construction Algorithm

Require: Graph $\mathcal{G}(V, E)$, node $j \in V$

1: $i \leftarrow 1$
2: $u_i \leftarrow j$
3: **while** $u_i \neq \mathtt{r}$ **do**
4: $\quad u_{i+1} \leftarrow \arg\min_{m \in \delta_{\text{in}}(u_i)} Q_{mu_i}(t)$;
5: $\quad i \leftarrow i + 1$
6: **end while**

$\sigma(\mathtt{r} \to j)$. The number of distinct packets received by node j up to time T can be written as a telescoping sum of relative packet deficits along the path σ,

$$\begin{aligned} R_j(T) &\equiv R_{u_1}(T) \\ &= \sum_{i=1}^{k-1} \big(R_{u_i}(T) - R_{u_{i+1}}(T)\big) + R_{u_k}(T) \\ &\overset{(*)}{=} -\sum_{i=1}^{k-1} X_{u_i}(T) + R_{\mathtt{r}}(T) \\ &= -\sum_{i=1}^{k-1} X_{u_i}(T) + \sum_{t=0}^{T-1} A(t), \end{aligned} \tag{2.53}$$

where the equality $(*)$ follows the observation that (see (2.10))

$$X_{u_i}(T) = Q_{u_{i+1}u_i}(T) = R_{u_{i+1}}(T) - R_{u_i}(T).$$

Using the bound $\sum_{i=1}^{k-1} X_{u_i}(t) \leq \sum_{j \neq \mathtt{r}} X_j(t)$ (since $X_j(t) \geq 0$) and Eqn. (3.40), we conclude that for every node $j \neq \mathtt{r}$,

$$\frac{1}{T}\sum_{t=0}^{T-1} A(t) - \frac{1}{T}\sum_{j \neq \mathtt{r}} X_j(T) \leq \frac{1}{T} R_j(T) \leq \frac{1}{T}\sum_{t=0}^{T-1} A(t).$$

Finally, using the Strong Law of Large Numbers for the arrival process $\{A(t)\}_{t \geq 0}$ and Eqn. (3.39), we conclude

$$\lim_{T \to \infty} \frac{R_j(T)}{T} = \lambda, \, \forall j. \quad \text{w.p. } 1$$

This concludes the proof.

2.9.2 Proof of Lemma 2.3.5

We regard the DAG $\mathcal{G}$ as a wire line network in which all links can be activated simultaneously at a slot. Theorem 2.3.4 and Eqn. (2.8) show that the broadcast capacity of the network $\mathcal{G}$ is

$$\begin{aligned}\lambda^* = \lambda_{\text{DAG}} = \min_{U\text{: a proper cut}} \sum_{e\in E_U} c_e &= \min_{\{U_v, v\neq \mathtt{r}\}} \sum_{e\in E_{U_v}} c_e \\ &\overset{(*)}{=} \min_{v\in V\setminus\{\mathtt{r}\}} d_{\text{in}}(v), \end{aligned} \tag{2.54}$$

where the sets U_v and E_{U_v} are defined in Eqns. (2.6) and (2.3) respectively. The equality $(*)$ follows from the assumption that $c_e = 1, \forall e \in E$. Edmond's Theorem [14] states that the maximum number of disjoint spanning trees in the graph $\mathcal{G}$ is

$$k^* = \min_{U\text{: a proper cut}} \sum_{e\in E_U} c_e. \tag{2.55}$$

Combining (2.54) and (2.55) completes the proof of the Lemma.

Chapter 3

Throughput-Optimal Broadcast on Time-Varying Wireless DAGs

3.1 Overview of the Results

In this chapter, we build upon the results of Chapter 2, and consider the problem of throughput-optimal broadcasting in a wireless DAG network with time-varying connectivity. Extending the results from the previous chapter, we characterize the broadcast capacity of time-varying wireless DAGs and propose an exact and an approximation algorithm to compute it efficiently. Next, we propose a dynamic link activation and packet scheduling policy, which obviates the need to maintain any global topological structures, such as spanning trees, yet achieves the capacity in a time-varying setting. In addition to throughput-optimality, the proposed algorithm enjoys the attractive property of *in-order* packet-delivery, which makes it particularly useful in various online applications, e.g., VoIP and live multimedia communication [41]. The proposed broadcast policy is *model-oblivious*, in the sense that its operation does not depend on detailed statistics of the random packet arrival or the network connectivity processes. We also show that the throughput-optimality of our algorithm is retained when the control decisions are made using *locally* available and possibly *delayed state information*.

The main technical contributions of this chapter are as follows:

- We define and characterize the broadcast capacity for wireless networks with time-varying connectivity. We show that the broadcast capacity of time-varying wireless directed acyclic networks can be computed efficiently in some settings. We then derive tight upper and lower bounds on broadcast capacity and utilize it to propose an efficient approximation algorithm to estimate the capacity in a general setting.

- We propose a throughput-optimal dynamic routing and scheduling policy for broadcasting in a wireless DAG with time-varying connectivity. This algorithm is of *Max-Weight* type and critically uses the idea of *in-order* packet delivery. To the best of our knowledge, this is the first throughput-optimal dynamic algorithm proposed for broadcasting in time-varying wireless networks.

- We extend our algorithm to the practical scenario when the nodes have access only to delayed state information. We show that the throughput-optimality of the policy is retained even when the rate of inter-node communication is made arbitrarily small.

- We illustrate our theoretical findings with extensive numerical simulations.

The rest of the chapter is organized as follows. Section 3.2 introduces the notion of time-variation into our usual static wireless network model. Section 3.3 defines and characterizes the broadcast capacity of a time-varying wireless DAG. It also provides an exact and an approximation algorithm to compute its broadcast capacity. Section 3.4 describes our capacity-achieving broadcast algorithm. Section 3.5 extends the algorithm to the setting of imperfect state information. Section 4.7 provides numerical simulation results to illustrate our theoretical findings. In section 6.7 we summarize our results and conclude this chapter.

3.2 Network Model

3.2.1 Model of Time-varying Wireless Connectivity

In this section, we incorporate time-variation into our basic static network model, described in Chapter 1. In a wireless network, the channel-SINRs vary with time because of fading, shadowing and node mobility [42]. To take this random variation into account, we consider a simple ON-OFF channel model, where at each slot an individual link can be in either one of the two states, ON and OFF. In the ON state, a link (i, j), if activated, can transmit c_{ij} packets per slot, while in the OFF state it can not transmit any packet [1]. In other words, at any slot, the entire network can be in any one configuration, out of finitely many possible configurations, denoted by the set Ξ. Each element $\sigma \in \Xi$ corresponds to a sub-graph $\mathcal{G}(V, E_\sigma) \subset \mathcal{G}(V, E)$, where $E_\sigma \subset E$ denotes the set of links that are ON at that slot. At every time-slot t, one of the configurations in the set Ξ is randomly realized. The network configuration at time t is represented by the vector $\boldsymbol{\sigma}(t) \in \{0, 1\}^{|E|}$, where

$$\boldsymbol{\sigma}(e, t) = \begin{cases} 1, & \text{if } e \in E_{\boldsymbol{\sigma}(t)} \\ 0, & \text{otherwise.} \end{cases}$$

At the time-slot t, the network controller can only activate a set of non-interfering links from the set $E_{\boldsymbol{\sigma}(t)}$ that are ON.

The network configuration $\{\boldsymbol{\sigma}(t)\}_{t\geq 1}$ evolves according to a stationary ergodic process with the stationary distribution $\{p(\sigma)\}_{\sigma\in\Xi}$ [43], where

$$\sum_{\sigma\in\Xi} p(\sigma) = 1, \;\; p(\sigma) > 0, \;\; \forall \sigma \in \Xi. \tag{3.1}$$

Since the underlying physical processes responsible for time-variation are often spatially-correlated [44], [45], the distribution of the link states is assumed to possess an arbitrary joint distribution. The detailed parameters of this process depend on the

[1] Generalization of the ON-OFF model, to a multi-level discretization of link-capacity is straightforward.

ambient physical environment, which is often difficult to measure. In particular, it is unrealistic to assume that the controller has knowledge of the statistical parameters of the process $\{\boldsymbol{\sigma}(t)\}_{t\geq 1}$. Fortunately, our proposed dynamic throughput-optimal broadcast policy does not require the statistical characterization of the configuration process or its stationary distribution $p(\boldsymbol{\sigma})$. This makes the policy robust and suitable for use in a dynamic setting.

Notations and Nomenclature:

In this section, we briefly discuss the notations and conventions used throughout the chapter. All vectors are assumed to be column vectors. For any set $\mathcal{X} \subset \mathbb{R}^k$, its convex-hull is denoted by $\text{conv}(\mathcal{X})$. Let $\big(U, V \setminus U\big)$ be a disjoint partition of the set of vertices of the graph, such that the source $\mathtt{r} \in U$ and $U \subsetneq V$. Such a partition will be called a *proper-partition*. To each proper partition corresponding to the node set U, associate the *proper-cut* vector $\boldsymbol{u} \in \mathbb{R}^m$, defined as follows:

$$\begin{aligned} \boldsymbol{u}_{i,j} &= c_{i,j} \quad \text{if } i \in U, j \in V \setminus U, (i,j) \in E \\ &= 0 \quad \text{otherwise} \end{aligned} \tag{3.2}$$

Denote the special, single-node proper-partitions by $U_j \equiv V \setminus \{j\}$, and the corresponding proper-cut vectors by $\boldsymbol{u}_j$, $\forall j \in V \setminus \{\mathtt{r}\}$. The set of all proper-cut vectors in the graph $\mathcal{G}$ is denoted by $\mathcal{U}$.

The *in-neighbour* set $\partial^{\text{in}}(j)$ of a node j is defined to be the set of all nodes $i \in V$ such that there is a directed edge $(i,j) \in E$. i.e.,

$$\partial^{\text{in}}(j) = \{i \in V : (i,j) \in E\} \tag{3.3}$$

Similarly, we define the *out-neighbour* set of a node j as

$$\partial^{\text{out}}(j) = \big\{i \in V : (j,i) \in E\big\} \tag{3.4}$$

For any two vectors $\boldsymbol{x}$ and $\boldsymbol{y}$ in $\mathbb{R}^m$, define the coordinate-wise product $\boldsymbol{z} \equiv \boldsymbol{x} \odot \boldsymbol{y}$ to be a vector in $\mathbb{R}^m$ such that $z_i = x_i y_i, 1 \leq i \leq m$.

For any set $\mathcal{S} \subset \mathbb{R}^m$ and any vector $\boldsymbol{v} \in \mathbb{R}^m$, the symbol $\boldsymbol{v} \odot \mathcal{S}$ denotes the set of all vectors obtained by the coordinate-wise product of the vector $\boldsymbol{v}$ and the elements of the set $\mathcal{S}$, i.e.,

$$\boldsymbol{v} \odot \mathcal{S} = \left\{ \boldsymbol{y} \in \mathbb{R}^m : \boldsymbol{y} = \boldsymbol{v} \odot \boldsymbol{s}, \boldsymbol{s} \in \mathcal{S} \right\} \tag{3.5}$$

The usual dot product between two vectors $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^m$ is defined as: $\boldsymbol{x} \cdot \boldsymbol{y} = \sum_{i=1}^{m} x_i y_i$.

3.3 Characterization of the Broadcast Capacity in a Time-Varying DAG

In this section, we derive a cut-set characterization of the broadcast capacity in a time varying DAG.

3.3.1 An Upper-bound on Broadcast Capacity

Consider a policy $\pi \in \Pi$ that achieves a broadcast rate of at least $\lambda^* - \epsilon$, for an $\epsilon > 0$. That such a policy exists, follows from the definition of the broadcast capacity λ^*.

Now consider any proper-cut U of the network $\mathcal{G}$. By the definition of a proper-cut, there exists a node $i \notin U$. Let $\boldsymbol{s}^\pi(t, \boldsymbol{\sigma}(t)) = (s_e^\pi(t, \boldsymbol{\sigma}(t)), e \in E)$ be the link activation vector chosen by policy π in slot t, upon observing the current network configuration $\boldsymbol{\sigma}(t)$. The maximum number of packets that can be transmitted across the cut U in slot t is upper bounded by the total capacity of all activated links across the cut-set U, given by $\sum_{e \in E_U} c_e s_e^\pi(t, \boldsymbol{\sigma}(t))$. Hence, the number of distinct packets received by node i by time T is at most the total available capacity across the cut U up to time T, subject to link activation decisions of the policy π. In other words,

$$R_i^\pi(T) \leq \sum_{t=1}^{T} \sum_{e \in E_U} c_e s_e^\pi(t, \boldsymbol{\sigma}(t)) = \boldsymbol{u} \cdot \sum_{t=1}^{T} \boldsymbol{s}^\pi(t, \boldsymbol{\sigma}(t)) \tag{3.6}$$

i.e.,

$$\frac{R_i^\pi(T)}{T} \leq \boldsymbol{u} \cdot \left(\frac{1}{T} \sum_{t=1}^{T} \boldsymbol{s}^\pi(t, \boldsymbol{\sigma}(t)) \right),$$

where the cut-vector $\boldsymbol{u} \in \mathbb{R}^m$, corresponds to the cut-set U, as in Eqn.(3.2). It follows that,

$$\begin{aligned} \lambda^* - \epsilon &\overset{(a)}{\leq} \min_{j \in V} \liminf_{T \to \infty} \frac{R_j^\pi(T)}{T} \leq \liminf_{T \to \infty} \frac{R_i^\pi(T)}{T} \\ &\leq \liminf_{T \to \infty} \boldsymbol{u} \cdot \left(\frac{1}{T} \sum_{t=1}^{T} \boldsymbol{s}^\pi(t, \boldsymbol{\sigma}(t)) \right), \end{aligned} \tag{3.7}$$

where the inequality (a) follows from the fact that π is a broadcast policy of rate at least $\lambda^* - \epsilon$. Since the above inequality holds for all proper-cuts $\boldsymbol{u}$, we have

$$\lambda^* - \epsilon \leq \min_{\boldsymbol{u} \in \mathcal{U}} \liminf_{T \to \infty} \boldsymbol{u} \cdot \left(\frac{1}{T} \sum_{t=1}^{T} \boldsymbol{s}^\pi(t, \boldsymbol{\sigma}(t)) \right) \tag{3.8}$$

The following technical lemma will prove to be useful for deriving an upper-bound on the broadcast capacity.

Lemma 3.3.1 *For any policy $\pi \in \Pi$, and any proper-cut vector $\boldsymbol{u}$, there exist a collection of vectors $\left(\boldsymbol{\beta}_\sigma^\pi \in \text{conv}(\mathcal{M}_\sigma)\right)_{\sigma \in \Xi}$, such that, the following holds* a.s.

$$\begin{aligned} &\min_{\boldsymbol{u} \in \mathcal{U}} \liminf_{T \to \infty} \boldsymbol{u} \cdot \left(\frac{1}{T} \sum_{t=1}^{T} \boldsymbol{s}^\pi(t, \boldsymbol{\sigma}(t)) \right) \\ &\qquad = \min_{\boldsymbol{u} \in \mathcal{U}} \boldsymbol{u} \cdot \left(\sum_{\sigma \in \Xi} p(\sigma) \boldsymbol{\beta}_\sigma^\pi \right) \end{aligned}$$

See Section 3.8.1 for the proof of this lemma. The above lemma essentially replaces the minimum cut-set bound of arbitrary activations in (3.8), by the minimum cut-set bound of a stationary randomized activation. Combining Lemma 3.3.1 with Eqn. (3.8), we conclude that for any policy $\pi \in \Pi$ of rate at least $\lambda^* - \epsilon$, there exists a collection of vectors $\{\boldsymbol{\beta}_\sigma^\pi \in \text{conv}(\mathcal{M}_\sigma)\}_{\sigma \in \Xi}$ such that

$$\lambda^* - \epsilon \leq \min_{\boldsymbol{u} \in \mathcal{U}} \boldsymbol{u} \cdot \left(\sum_{\sigma \in \Xi} p(\boldsymbol{\sigma}) \boldsymbol{\beta}_\sigma^\pi \right) \tag{3.9}$$

Maximizing the RHS of Eqn. (3.9) over all vectors $\left\{\boldsymbol{\beta}_\sigma \in \text{conv}(\mathcal{M}_\sigma),\, \sigma \in \Xi\right\}$ and letting $\epsilon \searrow 0$, we have the following universal upper-bound on the broadcast capacity λ^*

$$\lambda^* \leq \max_{\boldsymbol{\beta}_\sigma \in \text{conv}(\mathcal{M}_\sigma)} \min_{\boldsymbol{u} \in \mathcal{U}} \boldsymbol{u} \cdot \left(\sum_{\sigma \in \Xi} p(\boldsymbol{\sigma}) \boldsymbol{\beta}_\sigma \right) \tag{3.10}$$

Specializing the above bound for single-node cuts of the form $\boldsymbol{U}_j = (V \setminus \{j\}) \rightarrow \{j\}, \forall j \in V \setminus \{\mathtt{r}\}$, we have the following upper-bound

$$\lambda^* \leq \max_{\boldsymbol{\beta}_\sigma \in \text{conv}(\mathcal{M}_\sigma)} \min_{j \in V \setminus \{\mathtt{r}\}} \boldsymbol{u}_j \cdot \left(\sum_{\sigma \in \Xi} p(\boldsymbol{\sigma}) \boldsymbol{\beta}_\sigma \right) \tag{3.11}$$

It will be shown in Section 3.4 that in a DAG, our throughput-optimal policy π^* achieves a broadcast-rate equal to the RHS of the bound (3.11). In particular, we have the following theorem

Theorem 3.3.2 *The broadcast capacity λ^*_{DAG} of a time-varying wireless DAG is given by:*

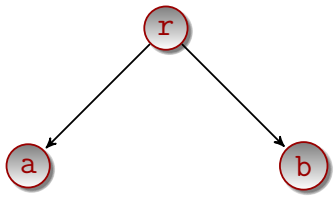
$$\lambda^*_{\text{DAG}} = \max_{\boldsymbol{\beta}_\sigma \in \text{conv}(\mathcal{M}_\sigma), \sigma \in \Xi} \min_{j \in V \setminus \{r\}} \boldsymbol{u}_j \cdot \left(\sum_{\sigma \in \Xi} p(\boldsymbol{\sigma}) \boldsymbol{\beta}_\sigma \right) \tag{3.12}$$

The above theorem shows that for computing the broadcast capacity of a wireless

DAG, the minimum in the bound in (3.10) is attained by the single-node cuts $\boldsymbol{u}_j$.

3.3.2 An Illustrative Example of Capacity Computation

In this section, we work out a simple example to illustrate the previous results.



Wireless network

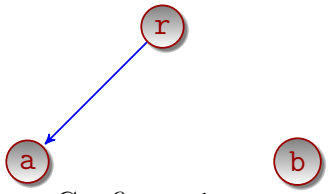
Consider the simple wireless network shown in Figure (3-1), with node `r` being the source. The possible network configurations $\sigma_i, i = 1, 2, 3, 4$ are also shown. One packet can be transmitted over a link if it is ON. Moreover, since the links are assumed to be point-to-point, even if both the links `ra` and `rb` are ON at a slot t (i.e., $\sigma(t) = \sigma_3$), a packet can be transmitted over one of the links only. Hence, the sets of feasible activations are given as follows:

$$\mathcal{M}_{\sigma_1} = \{\begin{pmatrix}1\\0\end{pmatrix}\}, \mathcal{M}_{\sigma_2} = \{\begin{pmatrix}0\\1\end{pmatrix}\}, \mathcal{M}_{\sigma_3} = \{\begin{pmatrix}1\\0\end{pmatrix}, \begin{pmatrix}0\\1\end{pmatrix}\}, \mathcal{M}_{\sigma_4} = \phi.$$

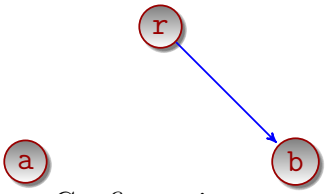
In the above vectors, the first coordinate corresponds to the edge `ra` and the second corresponds to the edge `rb`.

To illustrate the effect of link-correlations on broadcast capacity, we consider three different joint distributions $p(\boldsymbol{\sigma})$, all of them having the identical marginal:

$$p(\texttt{ra} = \text{ON}) = p(\texttt{ra} = \text{OFF}) = \frac{1}{2}$$
$$p(\texttt{rb} = \text{ON}) = p(\texttt{rb} = \text{OFF}) = \frac{1}{2}$$



Configuration σ_1



Configuration σ_2

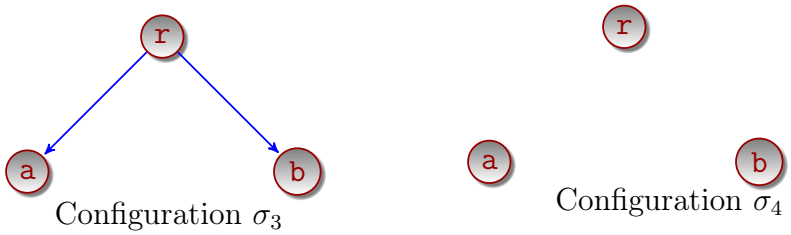


Figure 3-1: A Wireless Network and its four possible configurations

Case 1: Zero correlations In this case, the links `ra` and `rb` are ON w.p. $\frac{1}{2}$ independently at every slot, i.e.,

$$p(\sigma_i) = 1/4, \quad i = 1, 2, 3, 4 \tag{3.13}$$

It can be easily seen that the broadcast capacity, as given in Eqn. (3.12), is achieved when in configurations σ_1 and σ_2, the edges `ra` and `rb` are activated w.p. 1 respectively and in the configuration σ_3 the edges `ra` and `rb` are activated with probability $\frac{1}{2}$ and $\frac{1}{2}$. In other words, an optimal activation schedule of a corresponding stationary randomized policy is given as follows:

$$\boldsymbol{\beta}^*_{\sigma_1} = \begin{pmatrix} 1 & 0 \end{pmatrix}', \boldsymbol{\beta}^*_{\sigma_2} = \begin{pmatrix} 0 & 1 \end{pmatrix}', \boldsymbol{\beta}^*_{\sigma_3} = \begin{pmatrix} \frac{1}{2} & \frac{1}{2} \end{pmatrix}'$$

The optimal broadcast capacity can be computed from Eqn. (3.12) to be $\lambda^* = \frac{1}{4} + 0 + \frac{1}{4} \times \frac{1}{2} = \frac{3}{8}$.

Case 2: Positive correlations In this case, the edges `ra` and `rb` are positively correlated, i.e., we have

$$p(\boldsymbol{\sigma}_1) = p(\boldsymbol{\sigma}_2) = 0; \; p(\boldsymbol{\sigma}_3) = p(\boldsymbol{\sigma}_4) = \frac{1}{2}$$

Then it is clear that half of the slots are wasted when both the links are OFF (i.e., in the configuration $\boldsymbol{\sigma}_4$). When the network is in configuration $\boldsymbol{\sigma}_3$, an optimal randomized activation is to choose one of the two links uniformly at random and send

packets over it. Thus

$$\boldsymbol{\beta}^*_{\sigma_3} = (\frac{1}{2} \quad \frac{1}{2})'$$

The optimal broadcast capacity, as computed from Eqn. (3.12) is $\lambda^* = \frac{1}{4}$.

Case 3: Negative correlations In this case, the edges `ra` and `rb` are negatively correlated, i.e., we have

$$p(\boldsymbol{\sigma}_1) = p(\boldsymbol{\sigma}_2) = \frac{1}{2};\ p(\boldsymbol{\sigma}_3) = p(\boldsymbol{\sigma}_4) = 0$$

It is easy to see that, a capacity-achieving activation strategy, in this case, is to send packets over the link whichever is ON. The broadcast capacity in this case is $\lambda^* = \frac{1}{2}$, the highest among the above three cases.

As apparent from the above example, with an arbitrary joint distribution of network configurations $\{\boldsymbol{p}(\boldsymbol{\sigma})\}$, it is a matter of simple calculations to obtain the optimal activations $\boldsymbol{\beta}^*_{\boldsymbol{\sigma}}$ in Eqn. (3.12). However, it is clear that for an arbitrary network with arbitrary activations $\mathcal{M}$ and configuration sets Ξ, evaluating (3.12) is non-trivial. The following section deals with this computational problem.

3.3.3 Efficient Computation of the Broadcast Capacity

In this section, we study the problem of *efficient computation* of the Broadcast Capacity λ^* of a wireless DAG, given by Eqn. (3.12). In particular, we show that when the number of possible network configurations $|\Xi|(n)$ grows polynomially with n (the number of nodes in the network), there exists a strongly polynomial-time algorithm to compute λ^*, under the primary interference constraint. Polynomially-bounded network configurations arise, for example, when the set $\Xi(n)$ consists of subgraphs of the graph $\mathcal{G}$ with at most d number of edges, for some fixed integer d. In this case

$|\Xi(n)|$ can be bounded as follows

$$|\Xi|(n) \leq \sum_{k=0}^{d} \binom{m}{k} = \mathcal{O}(n^{2d}),$$

where $m(= \mathcal{O}(n^2))$ is the number of edges in the graph $\mathcal{G}$.

Theorem 3.3.3 (Efficient Computation of λ^*) *Suppose that for a wireless* DAG $\mathcal{G}$ *with n nodes, the number of possible network configurations $|\Xi|(n)$ is bounded polynomially in n. Then, there exists a strongly* $\mathsf{polynomial-time}$ *algorithm to compute the broadcast capacity of the network under the primary interference constraints.*

Although only polynomially many network configurations are allowed, we emphasize that Theorem (3.3.3) is highly non-trivial. This is because each network configuration $\sigma \in \Xi$ itself contains exponentially many possible activations (matchings) under the primary interference constraints. The key combinatorial result that leads to Theorem (3.3.3) is the existence of an efficient separator oracle for the matching-polytope for any arbitrary graph [38]. The detailed proof of Theorem 3.3.3 is provided below.

Proof Under the primary interference constraint, the set of feasible activations of the graphs are *matchings* [4]. To solve for the optimal broadcast capacity in a time-varying network, we first rewrite the optimization problem involved in Eqn. (3.12) as a Linear Program (LP). Although this LP has exponentially many constraints, using a well-known separation oracle for matchings, we show that it is possible to solve this LP in strongly polynomial time via the ellipsoid algorithm [37].

For a subset of edges $E' \subset E$, let $\chi^{E'}$ be the incidence vector, where $\chi^{E'}(e) = 1$ if

$e \in E'$ and is zero otherwise. Let

$$\mathcal{P}_{\text{matching}}(\mathcal{G}(V,E)) =$$
$$\textbf{convexhull}(\{\chi^M | M \text{ is a matching in } G(V,E)\})$$

We have the following classical result by Edmonds [38].

Theorem 3.3.4 *The set $\mathcal{P}_{matching}(\mathcal{G}(V,E))$ is characterized by the set of all $\boldsymbol{\beta} \in \mathbb{R}^{|E|}$ such that :*

$$\begin{aligned} \beta_e &\geq 0 \quad \forall e \in E \\ \sum_{e \in \partial^{\text{in}}(v) \cup \partial^{\text{out}}(v)} \beta_e &\leq 1 \quad \forall v \in V \\ \sum_{e \in E[U]} \beta_e &\leq \frac{|U|-1}{2}; \quad U \subset V,\ |U| \text{ odd} \end{aligned} \tag{3.14}$$

Here $E[U]$ is the set of edge with both end points in U.

Thus, following Eqn. (3.12), the broadcast capacity of a DAG can be obtained by the following LP :

$$\max \lambda \tag{3.15}$$

Subject to,

$$\lambda \leq \sum_{e \in \partial^{\text{in}}(v)} c_e \big(\sum_{\sigma \in \Xi} p(\sigma) \beta_{\sigma,e} \big),\ \forall v \in V \setminus \{r\} \tag{3.16}$$

$$\boldsymbol{\beta}_\sigma \in \mathcal{P}_{\text{matching}}(\mathcal{G}(V, E_\sigma)), \quad \forall \sigma \in \Xi \tag{3.17}$$

The constraint corresponding to $\sigma \in \Xi$ in (3.17) refers to the set of linear constraints given in Eqn.(3.14) corresponding to the graph $\mathcal{G}(V, E_\sigma)$, for each $\sigma \in \Xi$.

Invoking the equivalence of optimization and separation due to the ellipsoid algorithm

[37], it follows that the LP (3.15) is solvable in poly-time if there exists an efficient separator-oracle for the set of constraints (3.16) and (3.17). With our assumption of polynomially many network configurations $|\Xi|(n)$, there are only linearly many constraints ($n-1$, to be precise) in (3.16) with polynomially many variables in each constraint. Thus the set of constraints (3.16) can be separated efficiently. Next, we invoke a classic result from the combinatorial optimization literature which shows the existence of efficient separators for the matching polytopes.

Theorem 3.3.5 *[38] There exists a strongly poly-time algorithm, that given $\mathcal{G} = (V, E)$ and $\boldsymbol{\beta} : E \to \mathbb{R}^{|E|}$ determines if $\boldsymbol{\beta}$ satisfies* (3.14) *or outputs an inequality from* (3.3.4) *that is violated by $\boldsymbol{\beta}$.*

Hence, there exists an efficient separator for each of the constraints in (3.3.4). Since there are only polynomially many network configurations, this directly leads to Theorem 3.3.3.

3.3.4 Simple Bounds on λ^*

Using Theorem (3.3.3) we can, in principle, compute the broadcast capacity λ^* of any wireless DAG with polynomially many network configurations. However, the complexity of the exact computation of λ^* grows substantially with the number of the possible configurations $|\Xi|(n)$. Moreover, Theorem (3.3.3) does not apply when $|\Xi|(n)$ can no longer be bounded by a polynomial in n. A simple example with exponentially large $|\Xi|(n)$ is the case when any link e is ON w.p. $p_e > 0$ i.i.d. at every slot.

To address this issue, we obtain bounds on λ^*, whose computational complexity is independent of the size of $|\Xi|$. These bounds are conveniently expressed in terms of the broadcast capacity of the static network $\mathcal{G}(V, E)$ without time-variation, i.e. when $|\Xi| = 1$ and $E_\sigma = E, \sigma \in \Xi$. Let us denote the broadcast capacity of the static

network by λ^*_{stat}. Specializing Eqn. (3.12) to this case, we obtain

$$\lambda^*_{\text{stat}} = \max_{\boldsymbol{\beta} \in \text{conv}(\mathcal{M})} \min_{j \in V \setminus \{\mathtt{r}\}} \boldsymbol{u}_j \cdot \boldsymbol{\beta}. \tag{3.18}$$

Using Theorem (3.3.3), λ^*_{stat} can be computed in poly-time under the primary interference constraint.

Now consider an arbitrary joint distribution $p(\boldsymbol{\sigma})$ such that each link is ON uniformly with probability p, i.e.,

$$\sum_{\boldsymbol{\sigma} \in \Xi : \boldsymbol{\sigma}(e) = 1} p(\boldsymbol{\sigma}) = p, \quad \forall e \in E. \tag{3.19}$$

We have the following bounds on λ^* for this case:

Lemma 3.3.6 (Bounds on the Broadcast Capacity)

$$p\lambda^*_{stat} \leq \lambda^* \leq \lambda^*_{stat}.$$

Proof The proof consists of the following two parts:

Proof of the Upper-bound

Note that, for all $\sigma \in \Xi$, we have $E_\sigma \subset E$. Hence, it follows that

$$\mathcal{M}_\sigma \subset \mathcal{M}, \quad \forall \sigma \in \Xi$$

This, in turn, implies that

$$\beta_\sigma \in \text{conv}(\mathcal{M}_\sigma) \implies \beta_\sigma \in \text{conv}(\mathcal{M}) \tag{3.20}$$

Let an optimal solution to Eqn. (3.12) be obtained at $\big(\boldsymbol{\beta}^*_\sigma, \sigma \in \Xi\big)$. Then from Eqn. (3.20), it follows that

$$\sum_{\sigma\in\Xi} p(\boldsymbol{\sigma})\boldsymbol{\beta}^*_\sigma \in \text{conv}(\mathcal{M})$$

Hence we have,

$$\begin{aligned}\max_{\boldsymbol{\beta}_\sigma\in\text{conv}(\mathcal{M}_\sigma)} \min_{j\in V\setminus\{\mathtt{r}\}} \boldsymbol{u}_j \cdot \big(\sum_{\sigma\in\Xi} p(\boldsymbol{\sigma})\boldsymbol{\beta}^*_\sigma\big) \\ \leq \max_{\boldsymbol{\beta}\in\text{conv}(\mathcal{M})} \min_{j\in V\setminus\{\mathtt{r}\}} \boldsymbol{u}_j \cdot \boldsymbol{\beta}\end{aligned}$$

Using Eqn. (3.18), this shows that

$$\lambda^* \leq \lambda^*_{\text{stat}}$$

This proves the upper-bound.

Proof of the Lower-bound

Since $\mathcal{M}_\sigma \subset \mathcal{M}$, the expression for the broadcast capacity (3.12) may be re-written as follows:

$$\lambda^* = \max_{\boldsymbol{\beta}_\sigma\in\mathcal{M}} \min_{j\in V\setminus\{\mathtt{r}\}} \sum_{e\in\partial^{\text{in}}(j)} c_e\big(\sum_{\sigma\in\Xi} p(\boldsymbol{\sigma})\boldsymbol{\beta}_{\boldsymbol{\sigma}}(e)\mathbb{1}(e\in\boldsymbol{\sigma})\big)$$

Let $\boldsymbol{\beta}^* \in \text{conv}(\mathcal{M})$ be the optimal activation, achieving the RHS of (3.18). Hence we can lower-bound λ^* as follows

$$\begin{aligned}\lambda^* &\geq \min_{j\in V\setminus\{\mathtt{r}\}} \sum_{e\in\partial^{\text{in}}(j)} c_e\boldsymbol{\beta}^*(e)\big(\sum_{\sigma\in\Xi} p(\boldsymbol{\sigma})\mathbb{1}(e\in\boldsymbol{\sigma})\big) \\ &\overset{(a)}{=} p \min_{j\in V\setminus\{\mathtt{r}\}} \sum_{e\in\partial^{\text{in}}(j)} c_e\boldsymbol{\beta}^*(e) \\ &= p \min_{j\in V\setminus\{\mathtt{r}\}} \boldsymbol{u}_j\cdot\boldsymbol{\beta}^* \\ &\overset{(b)}{=} p\lambda^*_{\text{stat}}\end{aligned}$$

Equality (a) follows from the assumption (3.19) and equality (b) follows from the characterization (3.18). This proves the lower-bound.

Generalization of the above Lemma to the setting, where the links are ON with non-uniform probabilities, may also be obtained in a similar fashion.

More importantly, as the example 3.3.2 shows, the simple bounds in Lemma 3.3.6 are tight. In this example the value of the connectivity parameter $p = \frac{1}{2}$, the lower-bound is attained in case (2) and the upper-bound is attained in case (3).

The above lemma immediately leads to the following corollary:

Corollary 3.3.7 (Approximation-algorithm for computing λ^*). *Assume that, under the stationary distribution $p(\boldsymbol{\sigma})$, the probability that any link is ON is p, uniformly for all links. Then, there exists a poly-time p-approximation algorithm to compute the broadcast capacity λ^* of a* DAG*, under the primary interference constraints.*

Proof Consider the optimal randomized-activation vector $\boldsymbol{\beta}^* \in \text{conv}(\mathcal{M})$, corresponding to the stationary graph $\mathcal{G}(V, E)$ (3.18). By Theorem (3.3.3), $\boldsymbol{\beta}^*$ can be computed in poly-time under the primary interference constraint. Note that, by Caratheodory's theorem [39], the optimal $\boldsymbol{\beta}^*$ may be expressed as a convex combination of at most $|E| - 1$ matchings. Thus it follows that λ^*_{stat} (3.18) may also be computed in poly-time.

From the proof of Lemma 3.3.6, it follows that by randomly activating $\boldsymbol{\beta}^*$ (i.e., $\boldsymbol{\beta}_{\boldsymbol{\sigma}}(e) = \boldsymbol{\beta}^*(e)\mathbb{1}(e \in \sigma), \forall \boldsymbol{\sigma} \in \Xi$) we obtain a broadcast-rate equal to $p\lambda^*_{\text{stat}}$ where λ^*_{stat} is shown to be an upper-bound to the broadcast capacity λ^* in Lemma (3.3.6). Hence it follows that $p\lambda^*_{\text{stat}}$ constitutes a p-approximation to the broadcast capacity λ^*, which can be computed in poly-time.

This concludes our discussion the computational aspect of the broadcast capacity. In the rest of this chapter, we are concerned with designing a dynamic and throughput-optimal broadcast policy for a time-varying wireless DAG network.

3.4 Throughput-Optimal Broadcast Policy for Time-Varying Wireless DAGs

In this section, we propose an online, dynamic, throughput-optimal broadcast policy for time-varying wireless DAGs, that does not need to compute or maintain any global topological structures, such as spanning trees. Interestingly, we show that the broadcast-algorithm that we proposed in Chapter 2 for static wireless networks, generalizes well to the time-varying setting. As in the previous algorithm for the static network, this algorithm also enjoys the attractive operational property *in-order* packet delivery. The key difference between the algorithm in [46] and our dynamic algorithm is in link-scheduling. In particular, in our algorithm, the activation sets are chosen based on current network configuration $\boldsymbol{\sigma}(t)$.

3.4.1 Throughput-Optimal Broadcast Policy π^*

Any admissible broadcast policy $\pi \in \Pi$ comprise of the following two sub-modules that are executed at every slot t:

- $\pi(\mathcal{A})$ (**Activation module**): activates a subset of links $\boldsymbol{s}(t) \in \mathcal{M}_{\boldsymbol{\sigma}(t)}$, subject to the interference constraint and the current network configuration $\boldsymbol{\sigma}(t)$.
- $\pi(\mathcal{S})$ (**Packet Scheduling module**): schedules a subset of packets over the activated links.

Following our development in Chapter 2, we first restrict our attention to the policy sub-space $\Pi^{\text{in}-\text{order}}$, in which the admissible policies are required to follow the so-called *in-order* delivery property, defined as follows

Definition 3.4.1 (Policy-space $\Pi^{\text{in}-\text{order}}$) *A policy π belongs to the space $\Pi^{\text{in}-\text{order}}$ if all incoming packets are serially indexed as $\{1, 2, 3, \ldots\}$ according to their order of arrival at the source $\boldsymbol{r}$ and a node can receive a packet p at time t, only if it has already received the packets $\{1, 2, , \ldots, p-1\}$.*

As an immediate consequence of the *in-order* delivery property, the state of the received packets in the network at time-slot t may be succinctly represented by the n-dimensional vector $\boldsymbol{R}(t)$, whose i $^{\text{th}}$ component denotes the index of the *latest* packet received by node i by time t. We emphasize that this succinct network-state representation by the vector $\boldsymbol{R}(t)$ is valid only in the restricted policy-space $\Pi^{\text{in-order}}$. This compact state-representation results in substantial simplification of the overall state-space description. As a comparison, to completely specify the packet-configurations in the network at slot t in the general policy-space Π, we need to specify the *sets of packets* received by different nodes at slot t, which is quite unwieldy.
To effectively exploit the special structure of a DAG in designing our throughput-optimal broadcast policy, it will be useful to restrict our packet scheduler $\pi(\mathcal{S})$ further to the following policy-space $\Pi^* \subset \Pi^{\text{in-order}}$.

Definition 3.4.2 (Policy-space $\Pi^* \subset \Pi^{\text{in-order}}$) *A broadcast policy π belongs to the space Π^* if (1) $\pi \in \Pi^{\text{in-order}}$ and in addition (2) a packet p can be received by a node j at time t, only if all in-neighbours of the node j (i.e., nodes in $\partial^{\text{in}}(j)$) have received the packet p by the time t.*

The above definition is illustrated in Figure 3-2. The variables $X_j(t)$ and $i_t^*(j)$ appearing in its description are defined subsequently in Eqn. (3.23).

It is easy to see that for all policies $\pi \in \Pi^*$, the packet scheduler $\pi(\mathcal{S})$ is *completely*

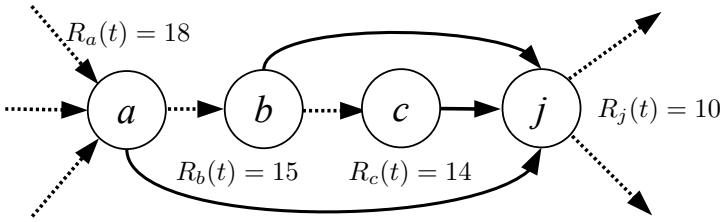


Figure 3-2: Under a policy $\pi \in \Pi^*$, the set of packets available for transmission to node j at slot t is $\{11, 12, 13, 14\}$, which are available at all in-neighbors of node j. The in-neighbor of j inducing the smallest packet deficit is $i_t^*(j) = c$, and $X_j(t) = 14 - 10 = 4$.

specified. Hence, to specify a policy in the space Π^*, we need to define the activation module $\pi(\mathcal{A})$ only.

Towards this end, let $\mu_{ij}(t)$ denote the rate (in packets per slot) allocated to the edge (i,j) in the slot t by a policy $\pi \in \Pi^*$. Note that, the allocated rate $\boldsymbol{\mu}(t)$ is constrained by the current network configuration $\boldsymbol{\sigma}(t)$ at slot t. In other words, we have

$$\boldsymbol{\mu}(t) \in \boldsymbol{c} \odot \mathcal{M}_{\boldsymbol{\sigma}(t)}. \tag{3.21}$$

This implies that, under any randomized activation

$$\mathbb{E}\boldsymbol{\mu}(t) \in \boldsymbol{c} \odot \mathrm{conv}(\mathcal{M}_{\boldsymbol{\sigma}(t)}). \tag{3.22}$$

In the following lemma, we show that for all policies $\pi \in \Pi^*$, certain state variables $\boldsymbol{X}(t)$, derived from the state-vector $\boldsymbol{R}(t)$, satisfy the *Lindley recursions* [47] of queueing theory. Hence these variables may be thought of as *virtual queues*. This technical result will play a central role in deriving a *Max-Weight* type throughput-optimal policy π^*, which is obtained by stochastically stabilizing these virtual-queues.
For each $j \in V \setminus \{\mathtt{r}\}$, define

$$X_j(t) = \min_{i \in \partial^{\mathrm{in}}(j)} \big(R_i(t) - R_j(t)\big) \tag{3.23}$$

$$i_t^*(j) = \arg\min_{i \in \partial^{\mathrm{in}}(j)} \big(R_i(t) - R_j(t)\big), \tag{3.24}$$

where in Eqn. (3.24), ties are broken lexicographically. The variable $X_j(t)$ denotes the minimum packet deficit of node j with respect to any of its in-neighbours. Hence, from the definition of the policy-space Π^*, it is clear that $X_j(t)$ is the maximum number of packets that a node j can receive from its in-neighbours at time t, under any policy in Π^*.
The following lemma, established in Chapter 2, proves a "queue-like-dynamics" of the variables $X_j(t)$, under any policy $\pi \in \Pi^*$.

Lemma 3.4.3 *Under any policy $\pi \in \Pi^*$, we have*

$$X_j(t+1) \leq \left(X_j(t) - \sum_{k \in \partial^{\text{in}}(j)} \mu_{kj}(t)\right)^+ + \sum_{m \in \partial^{\text{in}}(i_t^*(j))} \mu_{mi_t^*(j)}(t) \tag{3.25}$$

Lemma (3.4.3) shows that the variables $\big(X_j(t), j \in V \setminus \{\mathtt{r}\}\big)$ satisfy Lindley recursions in the policy-space Π^*. Interestingly, unlike the corresponding unicast problem [1], there is no "physical queue" in the system.

Similar to the unicast problem [1], the next lemma shows that any activation module $\pi(\mathcal{A})$ that "stabilizes" the *virtual queues* $\boldsymbol{X}(t)$ for all arrival rates $\lambda < \lambda^*$, constitutes a throughput optimal broadcast-policy for a wireless DAG network.

Lemma 3.4.4 *If under the action of a broadcast policy $\pi \in \Pi^*$, for all arrival rates $\lambda < \lambda^*$, the virtual queue process $\{\boldsymbol{X}(t)\}_0^\infty$ is rate-stable, i.e.,*

$$\limsup_{T \to \infty} \frac{1}{T} \sum_{j \neq \boldsymbol{r}} X_j(T) = 0, \ \text{ w.p. } 1,$$

then the policy $\pi \in \Pi^*$ is a throughput-optimal broadcast policy for a wireless DAG *network.*

Proof See Section 3.8.2.

Equipped with Lemma (3.4.4), we now set out to derive a dynamic activation module $\pi^*(\mathcal{A})$ to stabilize the virtual-queue process $\{\boldsymbol{X}(t)\}_0^\infty$ for all arrival rates $\lambda < \lambda^*$. Formally, the structure of the module $\pi^*(\mathcal{A})$ is defined by a mapping of the following form:

$$\pi^*(\mathcal{A}) : (\boldsymbol{X}(t), \boldsymbol{\sigma}(t)) \to \mathcal{M}_{\boldsymbol{\sigma}(t)}$$

Thus, the module $\pi^*(\mathcal{A})$ is stationary and dynamic as it depends on the current value of the state variables and the network configuration only. This activation module is different from the policy described in [46] as the latter is meant for static wireless networks and hence, does not take into account the time-variation of network configurations, which is the focus of this chapter.

To describe $\pi^*(\mathcal{A})$, we first define the following node-set

$$K_j(t) = \{m \in \partial^{\text{out}}(j) : j = i_t^*(m)\} \tag{3.26}$$

where the variables $i_t^*(m)$ are defined earlier in Eqn. (3.24). The activation module $\pi^*(\mathcal{A})$ is given in Algorithm 1. The resulting policy in the space Π^* with the activation module $\pi^*(\mathcal{A})$ is called π^*.

Algorithm 4 A Throughput-optimal Activation Module $\pi^*(\mathcal{A})$

1: To each link $(i,j) \in E$, assign a weight as follows:

$$W_{ij}(t) = \begin{cases} X_j(t) - \sum_{k \in K_j(t)} X_k(t), \text{ if } \boldsymbol{\sigma}_{(i,j)}(t) = 1 \\ 0, \quad \text{o.w.} \end{cases} \tag{3.27}$$

2: Select an activation $s^*(t) \in \mathcal{M}_{\boldsymbol{\sigma}(t)}$ as follows:

$$s^*(t) \in \arg \max_{\boldsymbol{s} \in \mathcal{M}_{\boldsymbol{\sigma}(t)}} \boldsymbol{s} \cdot \big(\boldsymbol{c} \odot \boldsymbol{W}(t)\big) \tag{3.28}$$

3: Allocate rates on the links as follows:

$$\boldsymbol{\mu}^*(t) = \boldsymbol{c} \odot \boldsymbol{s}^*(t) \tag{3.29}$$

Note that, in steps (1) and (2) above, the computation of link-weights and link activations depend explicitly on the current network configuration $\boldsymbol{\sigma}(t)$. As anticipated, in the following lemma, we show that the activation module $\pi^*(\mathcal{A})$ stochastically stabilizes the virtual-queue process $\{\boldsymbol{X}(t)\}_0^\infty$.

Lemma 3.4.5 *For all arrival rates $\lambda < \lambda^*$, under the action of the policy π^* in a DAG, the virtual-queue process $\{\boldsymbol{X}(t)\}_0^\infty$ is rate-stable, i.e.,*

$$\limsup_{T\to\infty} \frac{1}{T} \sum_{j\neq r} X_j(T) = 0, \text{ w.p. } 1$$

The proof of this lemma is based on a Lyapunov-drift argument [33]. Please refer to Section 3.8.3 for the complete proof.
Combining the lemmas (3.4.4) and (3.4.5), we immediately obtain the main result of this section:

Theorem 3.4.6 *The policy π^* is a throughput-optimal broadcast policy in a time-varying wireless* DAG *network.*

3.5 Throughput-Optimal Broadcasting with Infrequent Inter-node Communication

In practical mobile wireless networks, it is unrealistic to assume that every node has perfect network-wide packet state information at every slot. This is especially true in the case of time-varying dynamic networks, where network connectivity changes frequently. In this section, we extend the main results of section 3.4 by considering the setting where the nodes make control decisions with *imperfect* packet state information that they currently possess. We will show that the dynamic broadcast-policy π^* retains its throughput-optimality even in this scenario.

State-Update Model We assume that two nodes i and j can mutually update their knowledge of the set of packets received by the other node, only at those slots with

positive probability, when the corresponding wireless link (i, j) is in ON state. Otherwise, it continues working with the outdated packet state information. Throughout this section, we assume that the nodes have perfect information about the current network configuration $\boldsymbol{\sigma}(t)$.

Suppose that, the latest time prior to time t when the packet state update was made across the link (i, j) is $t - T_{(i,j)}(t)$. Here $T_{(i,j)}(t)$ is a random variable, supported on the set of non-negative integers. Assume that the network configuration process $\{\boldsymbol{\sigma}(t)\}_0^\infty$ evolves according to a finite-state, positive recurrent Markov-Chain, with the stationary distribution $\{p(\boldsymbol{\sigma}) > 0, \boldsymbol{\sigma} \in \Xi\}$. Using standard theory [48], it can be shown that the random variable $T(t) \equiv \sum_{(i,j)\in E} T_{(i,j)}(t)$ has bounded expectation for all time t .

Analysis of π^* with Imperfect Packet State Information Consider running the policy π^*, where each node j now computes the weights $W'_{ij}(t)$, given by Eqn.(3.27), of the incoming links $(i, j) \in E$, based on the latest packet state information available to it. In particular, for each of its in-neighbour $i \in \partial^{\text{in}}(j)$, the node j possess the following information of the number of packets received by node i:

$$R'_i(t) = R_i(t - T_{(ij)}(t)) \tag{3.30}$$

Now, if the packet scheduler module $\pi'(\mathcal{S})$ of a broadcast-policy π' takes scheduling decision based on the imperfect state information $\boldsymbol{R}'(t)$ (instead of the true state $\boldsymbol{R}(t)$), it still retains the following useful property:

Lemma 3.5.1 $\pi' \in \Pi^*$.

Proof Due to space constraints, the proof of this lemma has been included in the accompanying supplementary material.

The above lemma states that the policy π' inherits the in-order delivery property and the in-neighbour packet delivery constraint of the policy-space Π^*.

From Eqn. (3.27) it follows that, computation of link-weights $\{W_{ij}(t), i \in \partial^{\text{in}}(j)\}$ by node j requires packet state information of the nodes that are located within 2-hops from the node j. Thus, it is natural to expect that with an ergodic state-update process, the weights $W'_{ij}(t)$, computed from the imperfect packet state information, will not differ too much from the true weights $W_{ij}(t)$, on the average. Indeed, we can bound the difference between the link-weights $W'_{ij}(t)$, used by policy π' and the true link-weights $W_{ij}(t)$, as follows:

Lemma 3.5.2 *There exists a finite constant C such that, the expected weight $W'_{ij}(t)$ of the link (ij), locally computed by the node j using the random update process, differs from the true link-weight $W_{ij}(t)$ by at most C, i.e.*

$$|\mathbb{E}W'_{ij}(t) - W_{ij}(t)| \leq C \tag{3.31}$$

The expectation above is taken with respect to the random packet state update process.

Proof Due to space constraints, the proof of this lemma has been included in the accompanying supplementary material.

From lemma (3.5.2) it follows that the policy π', in which link-weights are computed using imperfect packet state information is also a throughput-optimal broadcast policy for a wireless DAG. Its proof is very similar to the proof of Theorem (3.4.6). However, since the policy π' makes scheduling decision using $\boldsymbol{W}'(t)$, instead of $\boldsymbol{W}(t)$, we need to appropriately bound the differences in drift using Lemma (3.5.2). The technical details are provided in Section 3.8.4.

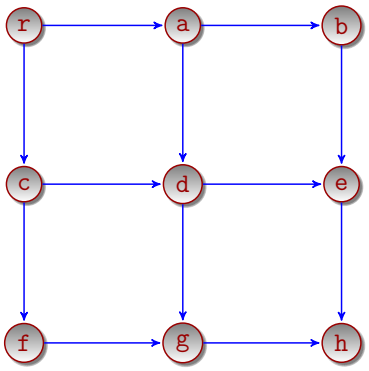


Figure 3-3: A 3×3 grid network.

Theorem 3.5.3 *The policy π' is a throughput-optimal broadcast algorithm in a time-varying wireless* DAG.

3.6 Numerical Simulation

We numerically simulate the performance of the proposed dynamic broadcast-policy on the 3×3 grid network, shown in Figure 3-3. All links are assumed to be of unit capacity. Wireless link activations are subject to primary interference constraints, i.e., at every slot, we may activate a subset of links which form a *Matching* [4] of the underlying topology. External packets arrive at the source node `r` according to a Poisson process of rate λ packets per slot. The following proposition shows that, the broadcast capacity λ^*_{stat} of the static 3×3 wireless grid (i.e., when all links are ON with probability 1 at every slot) is $\frac{2}{5}$.

Proposition 3.6.1 *The broadcast capacity λ^*_{stat} of the static 3×3 wireless grid network in Figure 3-3 is $\frac{2}{5}$.*

Due to space constraints, the proof of this proposition has been included in the

accompanying supplementary material.

In our numerical simulation, the time-variation of the network is modeled as follows: link states are assumed to evolving in an i.i.d. fashion; each link is ON with probability p at every slot, independent of everything else. Here $0 < p \leq 1$ is the *connectivity parameter* of the network. Thus, for $p = 1$ we recover the static network model of the previous chapter. We also assume that the nodes have imperfect packet state information as in Section 3.5. Two nodes i and j can directly exchange packet state information, only when the link (i, j) (if any) is ON.

The average broadcast delay $D_p^{\pi'}(\lambda)$ is plotted in Figure 3-4 as a function of the packet arrival rate λ. The broadcast delay of a packet is defined as the number of slots the packet takes to reach all nodes in the network after its arrival. Because of the throughput-optimality of the policy π' (Theorem (3.5.3)), the broadcast capacity $\lambda^*(p)$ of the network, for a given value of p, may be empirically evaluated from the λ-intercept of vertical asymptote of the $D_p^{\pi'}(\lambda) - \lambda$ curve.

As evident from the plot, for $p = 1$, the proposed dynamic algorithm achieves all broadcast rates below $\lambda^*_{\text{stat}} = \frac{2}{5} = 0.4$. This shows the throughput-optimality of the algorithm π'.

It is evident from the Figure 3-4 that the broadcast capacity $\lambda^*(p)$ is non-decreasing in the connectivity parameter p, i.e., $\lambda^*(p_1) \geq \lambda^*(p_2)$ for $p_1 \geq p_2$. We observe that, with i.i.d. connectivity, the capacity bounds given in Lemma (3.3.6) are not tight, in general. Hence the lower-bound of $p\lambda^*_{\text{stat}}$ is a pessimistic estimate of the actual broadcast capacity $\lambda^*(p)$ of the DAG. The plot also reveals that, $D_p^{\pi'}(\lambda)$ is non-decreasing in λ for a fixed p and non-increasing in p for a fixed λ, as expected.

Variation of the Broadcast Capacity with increasing network size

To investigate the variation of the broadcast capacity of a dynamic wireless network with network size, we numerically simulate the proposed throughput-optimal broadcast policy π^* on three wireless grid networks with grid sizes 5×5, 8×8 and 9×9

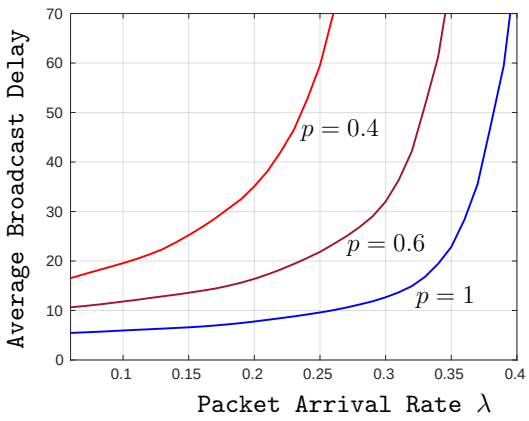


Figure 3-4: Plot of average broadcast delay $D_p^{\pi'}(\lambda)$, as a function of the packet arrival rates λ. The underlying wireless network is the 3×3 grid, shown in Figure 3-3, with primary interference constraints.

respectively. The edge-connectivity processes are assumed to be i.i.d. with the connectivity parameter $p = 0.8$ uniformly over all edges. All simulated networks are assumed to be limited by primary interference constraints.

The time-averaged broadcast delay as a function of the incoming packet arrival rate is plotted in Figure 3-5 for the three simulated networks. The λ-intercepts of the vertical asymptotes of the delay curves indicate the broadcast capacities $\lambda^*_{N\times N}$ of the corresponding grid networks. As expected, $\lambda^*_{N\times N}$ decreases with increasing network size.

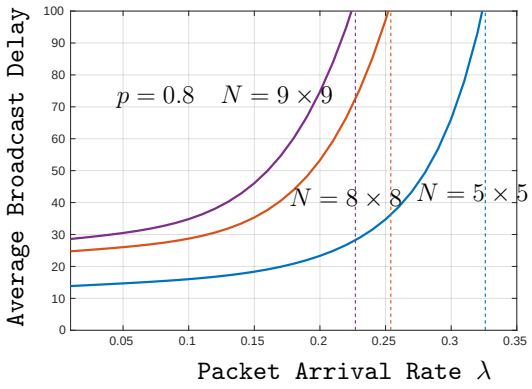


Figure 3-5: Plot of average broadcast delay $D_p^{\pi'}(\lambda)$, as a function of the packet arrival rates λ for i.i.d. connectivity process with parameter $p = 0.8$. Results are shown for 5×5, 8×8 and 9×9 grids with primary interference constraints. Vertical asymptotes indicate the respective broadcast capacities of the networks.

3.7 Conclusion and Future Work

In this chapter, we studied the problem of throughput-optimal broadcasting in wireless directed acyclic networks with point-to-point links and time-varying connectivity. We characterized the broadcast capacity of such networks and derived efficient algorithms for computing the same, both exactly and approximately. Next, we proposed a throughput-optimal broadcast policy for such networks. This policy does not need to maintain any spanning tree and operates based on locally available information, which is updated sporadically. The algorithm is robust and does not require statistics of the arrival or the connectivity process, thus making it useful for mobile wireless networks. The theoretical results are supplemented with illustrative numerical simulations. A possible future direction of research would be to remove the requirement of acyclic topology. It would also be interesting to extend the algorithm to wireless networks with point-to-multi-point links.

3.8 Proofs of the Results

3.8.1 Proof of Lemma 3.3.1

Proof Fix a time T. For each configuration $\sigma \in \Xi$, let $\{t_{\sigma,i}\}_{i=1}^{T_\sigma}$ be the index of the time-slots up to time T such that $\boldsymbol{\sigma}(t) = \boldsymbol{\sigma}$. Clearly, we have,

$$\sum_{\sigma \in \Xi} T_\sigma = T \tag{3.32}$$

Hence, we can rewrite

$$\frac{1}{T}\sum_{t=1}^{T} \boldsymbol{s}^\pi(t, \boldsymbol{\sigma}(t)) = \sum_{\sigma \in \Xi} \frac{T_\sigma}{T} \frac{1}{T_\sigma} \sum_{i=1}^{T_\sigma} \boldsymbol{s}^\pi(t_{\sigma,i}, \boldsymbol{\sigma}) \tag{3.33}$$

Hence,

$$\boldsymbol{u} \cdot \left(\frac{1}{T}\sum_{t=1}^{T} \boldsymbol{s}^\pi(t, \boldsymbol{\sigma}(t)) \right) = \sum_{\sigma \in \Xi} \frac{T_\sigma}{T} \boldsymbol{u} \cdot \left(\frac{1}{T_\sigma} \sum_{i=1}^{T_\sigma} \boldsymbol{s}^\pi(t_{\sigma,i}, \boldsymbol{\sigma}) \right) \tag{3.34}$$

Since the process $\boldsymbol{\sigma}(t)$ is stationary ergodic, we have

$$\lim_{T\to\infty} \frac{T_\sigma}{T} = p(\sigma), \text{ w.p. } 1 \ \forall \sigma \in \Xi \tag{3.35}$$

Using countability of Ξ and invoking the union bound, we can strengthen the above conclusion as follows

$$\lim_{T\to\infty} \frac{T_\sigma}{T} = p(\sigma), \ \forall \sigma \in \Xi, \text{ w.p. } 1 \tag{3.36}$$

Hence from Eqn. (3.34) we have,

$$\begin{aligned} & \min_{\boldsymbol{u}\in U} \liminf_{T\nearrow\infty} \boldsymbol{u} \cdot \left(\frac{1}{T} \sum_{t=1}^{T} \boldsymbol{s}^\pi(t, \boldsymbol{\sigma}(t)) \right) \\ = & \min_{\boldsymbol{u}\in U} \sum_{\sigma\in\Xi} p(\boldsymbol{\sigma}) \liminf_{T\to\infty} \boldsymbol{u} \cdot \left(\frac{1}{T_\sigma} \sum_{i=1}^{T_\sigma} \boldsymbol{s}^\pi(t_{\sigma,i}, \boldsymbol{\sigma}) \right), \quad \text{w.p. } 1 \end{aligned}$$

Since $p(\sigma) > 0, \forall \sigma \in \Xi$, the above implies that $T_\sigma \nearrow \infty$ as $T \nearrow \infty \forall \sigma$, w.p.1. In the rest of the proof, we will concentrate on a typical sample path $\{\boldsymbol{\sigma}(t)\}_{t\geq 1}$ having the above property.

For each $\sigma \in \Xi$, define the sequence $\{\boldsymbol{\zeta}^\pi_{\sigma,T_\sigma}\}_{T_\sigma \geq 1}$

$$\boldsymbol{\zeta}^\pi_{\sigma,T_\sigma} = \frac{1}{T_\sigma} \sum_{i=1}^{T_\sigma} \boldsymbol{s}^\pi(t_{\sigma,i}, \boldsymbol{\sigma}) \tag{3.37}$$

Since $\boldsymbol{s}^\pi(t_{\sigma,i}, \boldsymbol{\sigma}) \in \mathcal{M}_\sigma$ for all $i \geq 1$, convexity of the set $\mathcal{M}_\sigma$ implies that $\boldsymbol{\zeta}^\pi_{\sigma,T_\sigma} \in \mathcal{M}_\sigma$ for all $T_\sigma \geq 1$. Since the set $\mathcal{M}_\sigma$ is closed and bounded (and hence, compact) any sequence in $\mathcal{M}_\sigma$ has a converging sub-sequence. Consider any set of converging sub-sequences $\{\boldsymbol{\zeta}^\pi_{\sigma,T_{\sigma_k}}\}_{k\geq 1}, \sigma \in \Xi$ such that, it achieves the following

$$\begin{aligned} & \min_{u\in U} \sum_{\sigma\in\Xi} p(\sigma) \lim_{k\to\infty} \boldsymbol{u} \cdot \boldsymbol{\zeta}^\pi_{\sigma,T_{\sigma_k}} \\ = & \min_{u\in U} \sum_{\sigma\in\Xi} p(\sigma) \liminf_{T_\sigma\to\infty} \boldsymbol{u} \cdot \boldsymbol{\zeta}^\pi_{\sigma,T_\sigma}. \end{aligned}$$

Let us denote

$$\lim_{k\to\infty} \boldsymbol{\zeta}^{\pi}_{\sigma,T_{\sigma_k}} = \boldsymbol{\beta}^{\pi}_{\sigma}, \quad \forall \sigma \in \Xi \tag{3.38}$$

Where $\boldsymbol{\beta}^{\pi}_{\sigma} \in \mathcal{M}_{\sigma}$, since $\mathcal{M}_{\sigma}$ is closed. Hence combining Eqn. (3.37), (3.38) and Eqn. (3.38), we have

$$\begin{aligned} & \min_{\boldsymbol{u}\in U} \liminf_{T\nearrow\infty} \boldsymbol{u} \cdot \left(\frac{1}{T} \sum_{t=1}^{T} \boldsymbol{s}^{\pi}(t, \boldsymbol{\sigma}(t)) \right) \\ = & \min_{\boldsymbol{u}\in U} \sum_{\sigma\in\Xi} p(\boldsymbol{\sigma}) \boldsymbol{u} \cdot \boldsymbol{\beta}^{\pi}_{\sigma} \\ = & \min_{\boldsymbol{u}\in U} \boldsymbol{u} \cdot \left(\sum_{\sigma\in\Xi} p(\boldsymbol{\sigma}) \boldsymbol{\beta}^{\pi}_{\sigma} \right) \text{ w.p.1} \end{aligned}$$

3.8.2 Proof of Lemma (3.4.4)

Assume that under the policy $\pi \in \Pi^{*}$, the virtual queues $X_j(t)$ are rate stable i.e., $\lim_{T\to\infty} X_j(T)/T = 0, a.s.$ for all j. Applying union-bound, it follows that,

$$\lim_{T\to\infty} \frac{\sum_{j\neq r} X_j(T)}{T} = 0, \quad \text{w.p. 1} \tag{3.39}$$

Now consider any node $j \neq \mathtt{r}$ in the network. We can construct a simple path $p(\mathtt{r} = u_k \to u_{k-1} \ldots \to u_1 = j)$ from the source node $\mathtt{r}$ to the node j by running the following `Path construction algorithm` on the underlying graph $\mathcal{G}(V, E)$.

Algorithm 5 $\mathtt{r} \to j$ `Path Construction Algorithm`

Require: DAG $\mathcal{G}(V, E)$, node $j \in V$

1: $i \leftarrow 1$
2: $u_i \leftarrow j$
3: **while** $u_i \neq r$ **do**
4: $\quad u_{i+1} \leftarrow i^{*}_{t}(u_i)$;
5: $\quad i \leftarrow i + 1$
6: **end while**

At time t, the algorithm chooses the parent of a node u_i in the path p as the one that has the least relative packet deficit as compared to u_i (i.e. $u_{i+1} = i^{*}_{t}(u_i)$). Since the underlying graph $\mathcal{G}(V, E)$ is a connected DAG (i.e., there is a path from the

source to every other node in the network), the above path construction algorithm always terminates with a path $p(\mathtt{r} \to j)$. Note that the output path of the algorithm varies with time.

The number of distinct packets received by node j up to time T can be written as a telescoping sum of relative packet deficits along the path p, i.e.,

$$\begin{aligned} R_j(T) &= R_{u_1}(T) \\ &= \sum_{i=1}^{k-1} \big(R_{u_i}(T) - R_{u_{i+1}}(T)\big) + R_{u_k}(T) \\ &= -\sum_{i=1}^{k-1} X_{u_i}(T) + R_{\mathtt{r}}(T) \\ &\overset{(a)}{=} -\sum_{i=1}^{k-1} X_{u_i}(T) + \sum_{t=0}^{T-1} A(t), \end{aligned} \tag{3.40}$$

where the equality (a) follows the observation that

$$X_{u_i}(T) = Q_{u_{i+1}u_i}(T) = R_{u_{i+1}}(T) - R_{u_i}(T).$$

Since the variables $X_i(t)$'s are non-negative, we have $\sum_{i=1}^{k-1} X_{u_i}(t) \leq \sum_{j \neq r} X_j(t)$. Thus, for each node j

$$\frac{1}{T}\sum_{t=0}^{T-1} A(t) - \frac{1}{T}\sum_{j \neq r} X_j(T) \leq \frac{1}{T} R_j(T) \leq \frac{1}{T}\sum_{t=0}^{T-1} A(t).$$

Taking limit as $T \to \infty$ and using the strong law of large numbers for the arrival process and Eqn. (3.39), we have

$$\lim_{T\to\infty} \frac{R_j(T)}{T} = \lambda, \, \forall j. \quad \text{w.p. } 1$$

This concludes the proof.

3.8.3 Proof of Lemma (3.4.5)

We begin with a preliminary lemma.

Lemma 3.8.1 *If we have*

$$Q(t+1) \leq (Q(t) - \mu(t))^+ + A(t) \tag{3.41}$$

where all the variables are non-negative and $(x)^+ = \max\{x, 0\}$*, then*

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

Proof Squaring both sides of Eqn. (3.41) yields,

$$\begin{aligned}
&Q^2(t+1) \\
&\leq \big((Q(t) - \mu(t))^+\big)^2 + A^2(t) + 2A(t)(Q(t) - \mu(t))^+ \\
&\leq (Q(t) - \mu(t))^2 + A^2(t) + 2A(t)Q(t),
\end{aligned}$$

where we use the fact that $x^2 \geq (x^+)^2$, $Q(t) \geq 0$, and $\mu(t) \geq 0$. Rearranging the above inequality finishes the proof.

Applying Lemma 3.8.1 to the dynamics (3.25) of $X_j(t)$ yields, for each node $j \neq r$,

$$X_j^2(t+1) - X_j^2(t) \leq B(t) + \tag{3.42}$$

$$2X_j(t)\big(\sum_{m \in V} \mu_{mi_t^*}(t) - \sum_{k \in V} \mu_{kj}(t)\big), \tag{3.43}$$

where $B(t) \leq c_{\max}^2 + \max\{A^2(t), c_{\max}^2\} \leq (A^2(t) + 2c_{\max}^2)$, $A(t)$ is the number of exogenous packet arrivals in a slot, and $c_{\max} \triangleq \max_{e \in E} c_e$ is the maximum capacity of the links. Since per-slot arrival $A(t)$ has finite second moment, there exists a finite constant $B > 0$ such that $\mathbb{E}[B(t)] \leq \mathbb{E}\big(A^2(t)\big) + 2c_{\max}^2 < B$.

We define the quadratic Lyapunov function $L(\boldsymbol{X}(t)) = \sum_{j \neq r} X_j^2(t)$. From (3.42), the one-slot Lyapunov drift $\Delta(\boldsymbol{X}(t))$, *conditioned* on the current network configura-

tion $\boldsymbol{\sigma}(t)$ yields

$$\begin{aligned}
&\Delta(\boldsymbol{X}(t)|\boldsymbol{\sigma}(t)) \triangleq \mathbb{E}[L(\boldsymbol{X}(t+1) - L(\boldsymbol{X}(t)) \mid \boldsymbol{X}(t), \boldsymbol{\sigma}(t)] \\
= \;& \mathbb{E}\big[\sum_{j\neq r} \big(X_j^2(t+1) - X_j^2(t)\big) \mid \boldsymbol{X}(t), \boldsymbol{\sigma}(t)\big] \\
\leq \;& B|V| + 2\sum_{j\neq r} X_j(t)\mathbb{E}\big[\sum_{m\in V} \mu_{mi_t^*}(t) \qquad (3.44) \\
& - \sum_{k\in V} \mu_{kj}(t) \mid \boldsymbol{X}(t), \boldsymbol{\sigma}(t)\big] \\
= \;& B|V| - 2\sum_{(i,j)\in E} \mathbb{E}[\mu_{ij}(t) \mid \boldsymbol{X}(t), \boldsymbol{\sigma}(t)]\big(X_j(t) \qquad (3.45) \\
& - \sum_{k\in K_j(t)} X_k(t)\big) \\
= \;& B|V| - 2\sum_{(i,j)\in E} W_{ij}(t)\mathbb{E}[\mu_{ij}(t) \mid \boldsymbol{X}(t), \boldsymbol{\sigma}(t)] \qquad (3.46)
\end{aligned}$$

The broadcast-policy π^* is chosen to minimize the upper-bound of *conditional-drift*, given on the right-hand side of (3.46) among all policies in Π^*.

Next, we construct a randomized scheduling policy $\pi^{\text{RAND}} \in \Pi^*$. Let $\boldsymbol{\beta}^*_{\boldsymbol{\sigma}} \in \text{conv}(\mathcal{M}_\sigma)$ be the part of an optimal solution corresponding to $\boldsymbol{\sigma}(t) \equiv \boldsymbol{\sigma}$ given by Eqn. 3.10. From Caratheodory's theorem [39], there exist at most $(|E| + 1)$ link activation vectors $\boldsymbol{s}_k \in \mathcal{M}_\sigma$ and the associated non-negative scalars $\{\alpha_k^\sigma\}$ with $\sum_{k=1}^{|E|+1} \alpha_k^\sigma = 1$, such that

$$\boldsymbol{\beta}^*_\sigma = \sum_{k=1}^{|E|+1} \alpha_k^\sigma \boldsymbol{s}_k^\sigma. \qquad (3.47)$$

Define the average (unconditional) activation vector

$$\boldsymbol{\beta}^* = \sum_{\sigma\in\Xi} p(\sigma)\boldsymbol{\beta}^*_\sigma \qquad (3.48)$$

Hence, from Eqn. (3.10) we have,

$$\lambda^* \leq \min_{U:\text{ a proper cut}} \sum_{e\in E_U} c_e\beta_e^*. \qquad (3.49)$$

Suppose that the exogenous packet arrival rate λ is strictly less than the broadcast capacity λ^*. There exists an $\epsilon > 0$ such that $\lambda + \epsilon \leq \lambda^*$. From (3.49), we have

$$\lambda + \epsilon \leq \min_{U\text{: a proper cut}} \sum_{e \in E_U} c_e \beta_e^*. \tag{3.50}$$

For any network node $v \neq r$, consider the proper cuts $U_v = V \setminus \{v\}$. Specializing the bound in (3.50) to these cuts, we have

$$\lambda + \epsilon \leq \sum_{e \in E_{U_v}} c_e \beta_e^*, \ \forall v \neq r. \tag{3.51}$$

Since the underlying network topology $\mathcal{G} = (V, E)$ is a DAG, there exists a topological ordering of the network nodes so that: (i) the nodes can be labeled serially as $\{v_1, \ldots, v_{|V|}\}$, where $v_1 = r$ is the source node with no in-neighbours and $v_{|V|}$ has no outgoing neighbours and (ii) all edges in E are directed from $v_i \to v_j$, $i < j$ [40]; From (3.51), we define $q_l \in [0, 1]$ for each node v_l such that

$$q_l \sum_{e \in E_{U_{v_l}}} c_e \beta_e^* = \lambda + \epsilon \frac{l}{|V|}, \ l = 2, \ldots, |V|. \tag{3.52}$$

Consider the randomized broadcast policy $\pi^{\text{RAND}} \in \Pi^*$ working as follows:

Stationary Randomized Policy π^{RAND}:

(i) If the observed network configuration at slot t is $\boldsymbol{\sigma}(t) = \sigma$, the policy π^{RAND} **selects** [2] the feasible activation set $\boldsymbol{s}_k^\sigma$ with probability α_k^σ;

(ii) For each incoming selected link $e = (\cdot, v_l)$ to node v_l such that $s_e(t) = 1$, the link e is **activated** independently with probability q_l;

(iii) **Activated** links (note, not necessarily all the *selected* links) are used to forward packets, subject to the constraints that define the policy class Π^* (i.e., in-order packet delivery and that a network node is only allowed to receive packets that have been received by all of its in-neighbors).

Note that this stationary randomized policy π^{RAND} operates independently of the state of received packets in the network, i.e., $\boldsymbol{X}(t)$. However it depends on the current network configuration $\boldsymbol{\sigma}(t)$. Since each network node j is relabeled as v_l for some l, from (3.52) we have, for each node $j \neq r$, the total expected incoming transmission rate to the node j under the policy π^{RAND}, averaged over all network states $\boldsymbol{\sigma}$ satisfies

$$\begin{aligned}\sum_{i:(i,j)\in E} \mathbb{E}[\mu_{ij}^{\pi^{\mathrm{RAND}}}(t) \mid \boldsymbol{X}(t)] &= \sum_{i:(i,j)\in E} \mathbb{E}[\mu_{ij}^{\pi^{\mathrm{RAND}}}(t)] \\ &= q_l \sum_{e\in E_{U_{v_l}}} c_e \beta_e^* \\ &= \lambda + \epsilon \frac{l}{|V|}. \end{aligned} \tag{3.53}$$

Equation (3.53) shows that the randomized policy π^{RAND} provides each network node $j \neq r$ with the total expected incoming rate strictly larger than the packet arrival rate λ via proper random link activations conditioned on the current network configuration. According to our notational convention, we have

$$\sum_{i:(i,r)\in E} \mathbb{E}[\mu_{ir}^{\pi^{\mathrm{RAND}}}(t) \mid \boldsymbol{X}(t)] = \mathbb{E}[\sum_{i:(i,r)\in E} \mu_{ir}^{\pi^{\mathrm{RAND}}}(t)] = \lambda. \tag{3.54}$$

From (3.53) and (3.54), if node i appears before node j in the aforementioned topological ordering, i.e., $i = v_{l_i} < v_{l_j} = j$ for some $l_i < l_j$, then

$$\begin{aligned}&\sum_{k:(k,i)\in E} \mathbb{E}[\mu_{ki}^{\pi^{\mathrm{RAND}}}(t)] - \sum_{k:(k,j)\in E} \mathbb{E}[\mu_{kj}^{\pi^{\mathrm{RAND}}}(t)] \\ &\leq -\frac{\epsilon}{|V|}. \end{aligned} \tag{3.55}$$

The above inequality will be used to show the throughput optimality of the policy π^*.

The drift inequality (3.44) holds for any policy $\pi \in \Pi^*$. The broadcast policy π^* observes the states $(\boldsymbol{X}(t), \boldsymbol{\sigma}(t))$ and seek to *greedily* minimize the upper-bound of drift (3.46) at every slot. Comparing the actions taken by the policy π^* with those

by the randomized policy π^{RAND} in slot t in (3.44), we have

$$\begin{aligned}
&\Delta^{\pi^*}(\boldsymbol{X}(t)|\boldsymbol{\sigma}(t)) \qquad (3.56)\\
&\leq B|V| - 2\sum_{(i,j)\in E} \mathbb{E}\big[\mu_{ij}^{\pi^*}(t) \mid \boldsymbol{X}(t), \boldsymbol{\sigma}(t)\big] W_{ij}(t)\\
&\leq B|V| - 2\sum_{(i,j)\in E} \mathbb{E}\big[\mu_{ij}^{\pi^{\text{RAND}}}(t) \mid \boldsymbol{X}(t), \boldsymbol{\sigma}(t)\big] W_{ij}(t)\\
&\overset{(*)}{=} B|V| - 2\sum_{(i,j)\in E} \mathbb{E}\big[\mu_{ij}^{\pi^{\text{RAND}}}(t) \mid \boldsymbol{\sigma}(t)\big] W_{ij}(t)
\end{aligned}$$

(3.57)

Taking Expectation of both sides w.r.t. the stationary process $\boldsymbol{\sigma}(t)$ and rearranging, we have

$$\begin{aligned}
&\Delta^{\pi^*}(\boldsymbol{X}(t)) \qquad (3.58)\\
&\leq B|V| - 2\sum_{(i,j)\in E} \mathbb{E}\big[\mu_{ij}^{\pi^{\text{RAND}}}(t)\big] W_{ij}(t)\\
&\leq B|V| + 2\sum_{j\neq r} X_j(t)\Bigg(\sum_{m\in V} \mathbb{E}\big[\mu_{mi_t^*}^{\pi^{\text{RAND}}}(t)\big] - \sum_{k\in V} \mathbb{E}\big[\mu_{kj}^{\pi^{\text{RAND}}}(t)\big]\Bigg)\\
&\leq B|V| - \frac{2\epsilon}{|V|}\sum_{j\neq r} X_j(t). \qquad (3.59)
\end{aligned}$$

Note that $i_t^* = \arg\min_{i\in \text{In}(j)} Q_{ij}(t)$ for a given node j. Since node i_t^* is an in-neighbour of node j, i_t^* must lie before j in any topological ordering of the DAG. Hence, the last inequality of (3.59) follows directly from (3.55). Taking expectation in (3.59) with respect to $\boldsymbol{X}(t)$, we have

$$\mathbb{E}\big[L(\boldsymbol{X}(t+1))\big] - \mathbb{E}\big[L(\boldsymbol{X}(t))\big] \leq B|V| - \frac{2\epsilon}{|V|}\mathbb{E}||\boldsymbol{X}(t)||_1,$$

where $||\cdot||_1$ is the ℓ_1-norm of a vector. Summing the above inequality over $t =$

$0, 1, 2, \ldots T-1$ yields

$$\mathbb{E}\big[L(\boldsymbol{X}(T))\big] - \mathbb{E}\big[L(\boldsymbol{X}(0))\big] \leq B|V|T - \frac{2\epsilon}{|V|}\sum_{t=0}^{T-1}\mathbb{E}||\boldsymbol{X}(t)||_1.$$

Dividing the above by $2T\epsilon/|V|$ and using $L(\boldsymbol{X}(t)) \geq 0$, we have

$$\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}||\boldsymbol{X}(t)||_1 \leq \frac{B|V|^2}{2\epsilon} + \frac{|V|\,\mathbb{E}[L(\boldsymbol{X}(0))]}{2T\epsilon}$$

Taking a lim sup of both sides yields

$$\limsup_{T\to\infty}\frac{1}{T}\sum_{t=0}^{T-1}\sum_{j\neq r}\mathbb{E}[X_j(t)] \leq \frac{B|V|^2}{2\epsilon} \tag{3.60}$$

which implies that all virtual-queues $X_j(t)$ are strongly stable [33]. Strong stability of $X_j(t)$ implies that all virtual queues $X_j(t)$ are rate stable [33, Theorem 2.8].

3.8.4 Proof of Theorem (3.5.3)

To prove throughput-optimality of Theorem (3.5.3), we work with the same Lyapunov function $L(\boldsymbol{X}(t)) = \sum_{j\neq \mathbf{r}} X_j^2(t)$ as in Theorem (3.4.6) and follow the same steps until Eqn. (3.46) to obtain the following upper-bound on conditional drift

$$\begin{aligned} & \Delta^{\pi'}(\boldsymbol{X}(t)|\boldsymbol{X}(t), \boldsymbol{X}'(t), \boldsymbol{\sigma}(t)) \\ \leq\ & B|V| - 2\sum_{(i,j)\in E} W_{ij}(t)\mathbb{E}(\mu_{ij}^{\pi'}(t)|\boldsymbol{X}(t), \boldsymbol{X}'(t), \boldsymbol{\sigma}(t)) \end{aligned} \tag{3.61}$$

Since the policy π' makes scheduling decision based on the *locally computed* weights $W'_{ij}(t)$, by the definition of the policy π', we have for any policy $\pi \in \Pi$:

$$\begin{aligned} &\sum_{(i,j)\in E} W'_{ij}(t)\mathbb{E}(\mu^{\pi'}_{ij}(t)|\boldsymbol{X}(t),\boldsymbol{X}'(t),\boldsymbol{\sigma}(t)) \\ \geq &\sum_{(i,j)\in E} W'_{ij}(t)\mathbb{E}(\mu^{\pi}_{ij}(t)|\boldsymbol{X}(t),\boldsymbol{X}'(t),\boldsymbol{\sigma}(t)) \end{aligned} \tag{3.62}$$

Taking expectation of both sides w.r.t. the random update process $\boldsymbol{X}'(t)$, conditioned on the true network state $\boldsymbol{X}(t)$ and the network configuration $\boldsymbol{\sigma}(t)$, we have

$$\begin{aligned} & Cn^2 c_{\max}/2 + \sum_{(i,j)\in E} W_{ij}(t)\mathbb{E}(\mu^{\pi'}_{ij}(t)|\boldsymbol{X}(t),\boldsymbol{\sigma}(t)) \\ \overset{(a)}{\geq} & \sum_{(i,j)\in E} \mathbb{E}W'_{ij}(t)\mathbb{E}(\mu^{\pi'}_{ij}(t)|\boldsymbol{X}(t),\boldsymbol{\sigma}(t)) \\ \overset{(b)}{\geq} & \sum_{(i,j)\in E} \mathbb{E}W'_{ij}(t)\mathbb{E}(\mu^{\pi}_{ij}(t)|\boldsymbol{X}(t),\boldsymbol{\sigma}(t)) \\ \overset{(c)}{\geq} & \sum_{(i,j)\in E} W_{ij}(t)\mathbb{E}(\mu^{\pi}_{ij}(t)|\boldsymbol{X}(t),\boldsymbol{\sigma}(t)) - Cn^2 c_{\max}/2 \end{aligned} \tag{3.63}$$

Here the inequality (a) and (c) follows from Lemma (3.5.2) and the fact that $|E| \leq n^2/2$ and $\mu_{ij}(t) \leq c_{\max}$. Inequality (b) follows from Eqn. (3.62). Thus from Eqn. (3.61) and (3.63), the expected conditional drift of the Lyapunov function under the policy π', where the expectation is taken w.r.t. the random update and arrival process is upper bounded as follows:

$$\Delta^{\pi'}(\boldsymbol{X}(t)|\boldsymbol{X}(t),\boldsymbol{\sigma}(t)) \leq B' - 2\sum_{(i,j)\in E} W_{ij}(t)\mathbb{E}[\mu^{\pi}_{ij}(t) \mid \boldsymbol{X}(t),\boldsymbol{\sigma}(t)]$$

with the constant $B' \equiv B|V| + 2Cn^2 c_{\max}$. Since the above inequality holds for any policy $\pi \in \Pi$, we can follow the exactly same steps in the proof of Theorem (3.4.6) by replacing an arbitrary π by π^{RAND} and showing that it has negative drift.

Chapter 4

Throughput-Optimal Broadcast Algorithms : Networks with Arbitrary Topology

4.1 Overview of the Results

In this chapter, we drop the acyclicity assumption of Chapter 2 and 3, and address the problem of throughput-optimal broadcasting in arbitrary network topologies. To keep the algorithm and its analysis simple, we will propose and analyze a slightly different timing model (called the *minislot model*). However, this does not alter the essential nature of the problem and the broadcast algorithm that we design may be applied to the usual time-slotted model. The resulting broadcast policy has an interesting interpretation of executing "Backpressure on Sets". Our main technical contributions in this chapter are as follows:

(1) We first identify a convenient state-space representation of the network dynamics, in which the broadcast problem reduces to a "virtual-queue" stability problem, with appropriately defined "virtual queues". By utilizing Stochastic Lyapunov-drift techniques, we derive a broadcast policy that provably achieves the broadcast capacitt in arbitrary networks.

(2) Next, we introduce a multi-class heuristic policy, by combining the above policy with *in-class in-order* packet delivery from Chapter 2 in a suitable way. In this scheme, the number of classes is a tunable parameter, which offers a trade-off between efficiency and complexity. Several interesting properties of this heuristic scheme are also derived.

(3) Finally, we validate the theoretical ideas through extensive numerical simulations.

The rest of the chapter is organized as follows. In Section 4.2 we describe the operational network model (hereby referred to as the *minislot* model) and characterize its broadcast capacity. In Section 4.3 we derive our throughput-optimal broadcast policy. In Section 4.4 we propose a multi-class heuristic policy which uses the scheduling scheme derived in Section 4.3. Section 4.5 describes an extension of the policy to wireless networks, while Section 4.6 discusses distributed implementation of the proposed policy. In Section 4.7 we validate our theoretical results via numerical simulations. Finally, in section 4.8 we conclude this chapter with some directions for future work.

4.2 The Minislot Model

We begin our study with the consideration of broadcasting in wired networks with edge capacity constraints. This model is simpler to describe and analyze, yet it preserves all essential ingredients of the problem. The extension of the proposed broadcasting policy to wireless networks with activation constraints will be considered in Section 4.5.

As in the previous chapters, we consider a graph $\mathcal{G}(V, E)$, V being the set of vertices and E being the set of directed edges, with $|V| = n$ and $|E| = m$. In the equivalent slotted time model, the transmission capacity of each edge is one packet per slot. External packets arrive at the source node $\mathtt{r} \in V$. The arrivals are i.i.d. at every slot with expected arrival of λ packets per slot.

To simplify the analysis, we perturb the slotted-time assumption and adopt a slightly

different but equivalent *mini-slot* model. A slot consists of m consecutive mini-slots. Our dynamic broadcast algorithms are conceptually easier to derive, analyze and understand in the mini-slot model. However, the resulting algorithms can be easily adapted to the usual slotted model.

Mini-slot model: In this model, the basic unit of time is called a *mini-slot*. At each mini-slot t, an edge $e = (a, b) \in E$ is chosen for activation, independently and uniformly at random from the set of all m edges. All other $m - 1$ edges remain idle for that mini-slot. A packet can be transmitted over an active edge only. A single packet transmission takes one mini-slot for completion. This random edge-activity process is represented by the i.i.d. sequence of random variables $\{S(t)\}_{t=1}^{\infty}$, such that, $S(t) = e$ indicates that the edge $e \in E$ is activated at the mini-slot t. Thus,

$$\mathbb{P}(S(t) = e) = 1/m, \quad \forall e \in E, \ \forall t$$

External packets arrive at the source `r` with expected arrival of λ/m packets per mini-slot.

The main analytical advantage of the mini-slot model is that only a single packet transmission takes place at a mini-slot, which makes it easier to express the system-dynamics. Moreover, we will show in Theorem (4.2.3) that the broadcast capacity is the same in the two models.

4.2.1 Broadcast-Capacity of a Network

Informally, a network supports a broadcast rate λ if there exists a scheduling policy, under which all nodes in the network receive packets at the rate of λ, for the same rate of packet arrival at the source. The broadcast-capacity λ^* is the maximally achievable broadcast rate in the network.

In the minislot model, we consider a class Π of scheduling policies, which observe the currently active edge $e = (a, b)$ at every mini-slot t and select at most one packet from node a and transmit it to b over the active edge e. On the other hand, in the slotted time model, admissible policies in Π may transmit at most one packet per

edge *simultaneously* across all edges in the network at every slot. The policy-class Π includes policies that have access to all past and future information, and may forward any packet present at node a at time t to node b.

Recall that, a slot corresponds to m consecutive mini-slots. In either model, let $R^\pi(T)$ be the number of distinct packets received in common by all nodes in the network, up to slot T, under a policy $\pi \in \Pi$. The time average $\lim_{T\to\infty} R^\pi(T)/T$ is the rate at which packets are received uniformly at all nodes.

Definition 4.2.1 *A policy $\pi \in \Pi$ achieves a broadcast throughput λ, if for a packet arrival rate of λ, we have*

$$\lim_{T\to\infty} \frac{1}{T} R^\pi(T) = \lambda, \quad \text{in probability.} \tag{4.1}$$

Definition 4.2.2 *The broadcast capacity λ^* of a network is the supremum of all arrival rates λ for which there exists a broadcast policy $\pi \in \Pi$, achieving rate λ.*

A policy, that achieves any rate $\lambda < \lambda^*$, is called a throughput-optimal policy. In the slotted-time model, the broadcast capacity λ^* of a network $\mathcal{G}$ follows from the Edmonds' tree-packing theorem [14], and is given by the following:

$$\lambda^* = \min_{\mathtt{t}\in V\setminus\{\mathtt{r}\}} \text{Max-Flow}(\mathtt{r} \to \mathtt{t}) \quad \text{per slot}, \tag{4.2}$$

where Max-Flow($\mathtt{r} \to \mathtt{t}$) denotes the maximum value of *flow* that can be feasibly sent from the node $\mathtt{r}$ to the node $\mathtt{t}$ in the graph $\mathcal{G}(V, E)$ [49]. Edmonds' tree-packing theorem also implies that there exist λ^* edge-disjoint arborescences [1] or directed spanning trees, rooted at $\mathtt{r}$ in the graph $\mathcal{G}$. By examining the flow from the source to every node and using (4.2), it follows that by sending unit flow over each edge-disjoint tree, we may achieve the capacity λ^*.

As an illustration, consider the graph shown in Figure 4-1. It follows from Eqn. (4.2) that the broadcast capacity of the graph is $\lambda^* = 2$. Edges belonging to a set of two

[1] An *arborescence* is a directed graph such that there is a unique directed path from the root to all other vertices in it. Thus, an arborescence is a directed spanning tree. From now onwards, the terms "arborescence" and "directed spanning tree" will be used interchangeably.

edge-disjoint spanning trees $\mathcal{T}_1$ and $\mathcal{T}_2$ are shown in blue and red in the figure.
The following theorem establishes the equivalence of the *mini-slot* model and the *slotted-time* model in terms of broadcast capacity.

Theorem 4.2.3 (Invariance of Capacity) *The broadcast capacity of the mini-slot model is the same as that of the slotted-time model and is given by Eqn.* (4.2).

Proof See Appendix 4.9.1.

4.3 A Throughput-Optimal Broadcast Policy π^*

In this section we design a throughput-optimal broadcast policy $\pi^* \in \Pi$, for networks with arbitrary topology. This algorithm is of *Max-weight* type and is inspired by the seminal back-pressure policy for the corresponding unicast problem [1]. However, because of packet duplications, the usual per-node queues cannot be defined here. We get around this difficulty by defining certain virtual-queues, corresponding to subsets of nodes. We show that a scheduling policy in Π, which *stochastically stabilizes* these virtual queues for all arrival rates $\lambda < \lambda^*$, constitutes a throughput-optimal broadcast policy. Based on this result, we derive a Max-Weight policy π^*, by minimizing the drift of a quadratic Lyapunov function of the virtual queues.

4.3.1 Definitions and Notations

To facilitate the description of our proposed algorithm, we first introduce the notion of *reachable sets* and *reachable sequence of sets* as follows:

Definition 4.3.1 (Reachable Set) *A subset of vertices $F \subset V$ is said to be* reachable *if the induced graph* [2] *$F(\mathcal{G})$ contains a directed arborescence, rooted at source $\mathtt{r}$, which spans the node set F.*

[2]For a graph $\mathcal{G}(V, E)$ and any vertex set $F \subset V$, the induced graph $F(\mathcal{G})$ is defined as the sub-graph containing only the vertices F with the edges whose both ends lie in the set F.

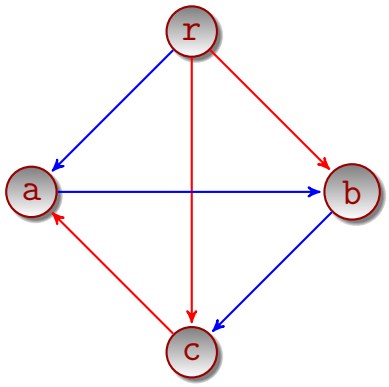


Figure 4-1: The four-node diamond network $\mathcal{D}_4$.

Equivalently, a subset of vertices $F \subset V$ is reachable if and only if there is a broadcast policy under which a packet p can be duplicated exactly in the subset F, during its course of broadcast. Note that, the set of all reachable sets is a strict subset of the set of all subsets of vertices. This is true because all reachable sets, by definition, must contain the source node `r`.

We may completely specify the *trajectory* of a packet during its course of broadcast, using the notion of *Reachable Sequences*, defined as follows:

Definition 4.3.2 (Reachable Sequence) *An ordered sequence of $n-1$ (reachable set, edge) tuples $\{(F_j, e_j), j = 1, 2, \ldots, n-1\}$ is called a* Reachable Sequence *if the following properties hold:*

- $F_1 = \{\mathtt{r}\}$ *and for all* $j = 1, 2, \ldots, n-1$*:*
- $F_j \subset F_{j+1}$
- $|F_{j+1}| = |F_j| + 1$.
- $e_j = (a, b) \in E : a \in F_j, b \in F_{j+1} \setminus F_j$

$\mathcal{F}$ is defined to be the set of all reachable sequences.

A reachable sequence denotes a feasible sequence of transmissions for broadcasting a particular packet to all nodes, where the j^{th} transmission of a packet takes place

across the edge $e_j, j = 1, 2 \ldots, n-1$. By definition, every reachable set must belong to at least one reachable sequence. A trivial upper-bound on $|\mathcal{F}|$ is n^{2n}. An example illustrating the notions of reachable sets and reachable sequences for a simple graph is provided next.

Example: Consider the graph shown in Figure 4-1. A reachable sequence for this graph is given by $\mathcal{S}$ below:

$$\mathcal{S} = \{(\{\texttt{r}\}, \texttt{rc}), (\{\texttt{r,c}\}, \texttt{ca}), (\{\texttt{r,a,c}\}, \texttt{rb})\}$$

This reachable sequence is obtained by adding nodes along the tree with red edges in Figure 4-1. Clearly, an example of a reachable set F in this graph is

$$F = \{\texttt{r,a,c}\}$$

For a reachable set F, define its set of out-edges $\partial^+ F$ and in-edges $\partial^- F$ as follows:

$$\partial^+ F \equiv \big\{(a,b) \in E : a \in F, b \notin F\big\} \tag{4.3}$$

$$\partial^- F \equiv \big\{(a,b) \in E : a \in F, b \in F\big\} \tag{4.4}$$

For an edge $e = (a,b) \in \partial^+ F$, define

$$F + e \equiv F \cup \{b\} \tag{4.5}$$

Similarly, for an edge $e = (a,b) \in \partial^- F$, define

$$F \setminus \{e\} \equiv F \setminus \{b\} \tag{4.6}$$

Convergence of Random Variables: For a sequence of random variables $\{X_n\}_{n=1}^{\infty}$ and another random variable X, defined on the same probability space, by the notation $X_n \overset{p}{\Longrightarrow} X$ we mean that the sequence of random variables $\{X_n\}_{n=1}^{\infty}$ converges *in*

probability to the random variable X [50].

4.3.2 System Dynamics

Consider any broadcast policy $\pi \in \Pi$ in action. For any reachable set $F \subsetneq V$, denote the number of packets, replicated *exactly* at the vertex-set F at mini-slot t, by $Q_F(t)$[3]. A packet p, which is replicated exactly at the set F by time t, is called a *class-F packet*. Hence, at a given time t, the reachable sets $F \in \mathcal{F}$ induce a disjoint partition of all the packets present in the network.

In our mini-slot model, a class-F packet can make a transition only to class $F + e$ (where $e \in \partial^+ F$) during a mini-slot, where e is the active edge. Let the rate allocated to the edge e, for transmitting a class-F packet at time t, be denoted by $\mu_{e,F}(t)$ (naturally, $\mu_{e,F}(t) \equiv 0$, if F is not a reachable set or e is inactive) [4]. Here $\mu_{e,F}(t)$ is a binary-valued control variable, which assumes the value 1 if the active edge e is allocated to transmit a class-F packet at the mini-slot t.

In the following we argue that, for any reachable set F, the variable $Q_F(t)$ satisfies the following one-step queuing-dynamics (Lindley recursion) [47]:

$$Q_F(t+1) \leq \left(Q_F(t) - \sum_{e \in \partial^+ F} \mu_{e,F}(t)\right)^+ + \sum_{(e,G): e \in \partial^- F, G = F \setminus \{e\}} \mu_{e,G}(t), \; \forall F \neq \{\mathtt{r}\} \quad (4.7)$$

$$Q_{\{\mathtt{r}\}}(t+1) \leq \left(Q_{\{\mathtt{r}\}}(t) - \sum_{e \in \partial^+(\{\mathtt{r}\})} \mu_{e,\{\mathtt{r}\}}(t)\right)^+ + A(t) \quad (4.8)$$

The dynamics in Eqn. (4.7) may be derived as follows: in the mini-slot model, only one packet over the currently active edge can be transmitted in the entire network at any mini-slot. Hence, for any reachable set F, the value of the corresponding state-variable $Q_F(t)$ may go up or down by at most one in a mini-slot. Now, $Q_F(t)$ decreases by one when any of the out-edges $e \in \partial^+ F$ is activated at mini-slot t and it carries a class-F packet, provided $Q_F(t) > 0$. This explains the first term in

[3] In the rest of this chapter, we define $Q_V(t) = 0, \forall t$.

[4] Note that $\mu_{e,F}(t)$ and consequently, $Q_F(t)$ depend on the used policy π and should be denoted by $\mu^{\pi}_{e,F}(t)$ and $Q^{\pi}_F(t)$. Here we drop the superscript π to simplify notation.

Eqn. (4.7). Similarly, the variable $Q_F(t)$ increases by one when a packet in some set $G = F \setminus \{e\}$ (or an external packet, in case $F = \{\mathtt{r}\}$), is transmitted to the set F over the (active) edge $e \in \partial^- F$. This explains the second term in Eqn. (4.7). In the following, we slightly abuse the notation by setting $\sum_{(e,G):e\in\partial^- F, G=F\setminus\{e\}} \mu_{e,G}(t) \equiv A(t)$, when $F = \{\mathtt{r}\}$. With this convention, the system dynamics is completely specified by the first inequality in (4.7), which constitutes a discrete time **Lindley recursion** [47].

4.3.3 Relationship between Stability and Throughput Optimality

The following lemma shows that stability of the virtual queues implies throughput-optimality for any admissible policy.

Lemma 4.3.3 (Stability implies Throughput-Optimality) *Consider a Markovian policy π, under which the induced Markov Chain $\{\boldsymbol{Q}^\pi(t)\}_0^\infty$ is Positive Recurrent for all arrival rate $\lambda < \lambda^*$. Then π is a throughput optimal broadcast policy.*

Proof Under the action of a Markovian Policy π, the total number of packets $R^\pi(T)$ delivered to all nodes in the network by the time T is given by

$$R^\pi(T) = \sum_{t=1}^{T} A(t) - \sum_F Q_F^\pi(T)$$

Hence, the rate of packet broadcast is given by

$$\begin{aligned}
\lim_{T\to\infty} \frac{R^\pi(T)}{T} &= \lim_{T\to\infty} \left(\frac{1}{T} \sum_{t=1}^{T} A(t) - \sum_F \frac{Q_F^\pi(T)}{T} \right) \\
&\overset{p}{\Longrightarrow} \lambda - \sum_F \lim_{T\to\infty} \frac{Q_F^\pi(T)}{T} \qquad (4.9) \\
&\overset{p}{\Longrightarrow} \lambda \qquad (4.10)
\end{aligned}$$

Eqn. (4.9) follows from the Weak Law of Large Numbers for the arrival process. To justify Eqn. (4.10), note that for any $\delta > 0$ and any reachable set F, we have

$$\lim_{T\to\infty} \mathbb{P}\left(\frac{Q_F^{\pi}(T)}{T} > \delta\right) = \lim_{T\to\infty} \mathbb{P}\left(Q_F^{\pi}(T) > T\delta\right) = 0, \tag{4.11}$$

where the last equality follows from the assumption of positive recurrence of $\{\boldsymbol{Q}^{\pi}(t)\}$. Thus Eqn. (4.11) implies that $\frac{Q_F^{\pi}(T)}{T} \stackrel{p}{\Longrightarrow} 0, \forall F$. This justifies Eqn. (4.10) and proves the lemma.

4.3.4 Stochastic Stability of the Process $\{\mathbf{Q}(t)\}_{t\geq 1}$

Equipped with Lemma (4.3.3), we now focus on finding a Markovian policy π^*, which stabilizes the chain $\{\boldsymbol{Q}^{\pi^*}(t)\}_{t\geq 1}$[5]. To accomplish this goal, we use the Lyapunov drift methodology [33], and derive a dynamic policy π^* which minimizes the one-minislot drift of a certain Lyapunov function. We then show that the proposed policy π^* has negative drift outside a bounded region in the state-space. Upon invoking the Foster-Lyapunov criterion [51], this proves positive recurrence of the chain $\{\boldsymbol{Q}(t)\}_0^{\infty}$.
To apply the scheme outlined above, we start out by defining the following Quadratic Lyapunov Function $L(\boldsymbol{Q}(t))$:

$$L(\boldsymbol{Q}(t)) = \sum_F Q_F^2(t), \tag{4.12}$$

where the sum extends over all reachable sets. Recall that, the r.v. $S(t)$ denotes the currently active edge at the mini-slot t. The one-minislot drift is defined as:

$$\Delta_t(\boldsymbol{Q}(t), S(t)) \equiv L(\boldsymbol{Q}(t+1)) - L(\boldsymbol{Q}(t)) \tag{4.13}$$

From the dynamics (4.7), we have

$$Q_F^2(t+1) \leq Q_F^2(t) + \mu_{\max}^2 - 2Q_F(t)\left(\sum_{e\in\partial^+ F} \mu_{e,F}(t) - \sum_{(e,G):e\in\partial^- F, G=F\setminus\{e\}} \mu_{e,G}(t)\right),$$

[5]The argument t denotes time in mini-slots.

where $\mu_{\max} = 1$ is the maximum capacity of a link per mini-slot. Thus, one mini-slot drift may be upper-bounded as follows:

$$\Delta_t(\boldsymbol{Q}(t), S(t)) \leq 2^n \mu_{\max}^2 - 2 \sum_{F \subsetneq V} Q_F(t) \Bigg(\sum_{e \in \partial^+ F} \mu_{e,F}(t) - \sum_{(e,G): e \in \partial^- F, G = F \setminus \{e\}} \mu_{e,G}(t) \Bigg).$$

Interchanging the order of summation, we have

$$\Delta_t(\boldsymbol{Q}(t), S(t)) \leq 2^n \mu_{\max}^2 - \sum_{(e,F): e \in \partial^+ F} \mu_{e,F}(t) \Big(Q_F(t) - Q_{F+e}(t) \Big).$$

Taking expectation of both sides of the above inequality with respect to the edge-activation process $S(t)$ and the arrival process $A(t)$, we obtain the following upper-bound on the conditional Lyapunov drift $\Delta_t(\boldsymbol{Q}(t))$:

$$\begin{aligned} \Delta_t(\boldsymbol{Q}(t)) &\equiv \mathbb{E}_{S(t)} \Delta_t(\boldsymbol{Q}(t), S(t)) \\ &\leq 2^n \mu_{\max}^2 - \sum_{(e,F): e \in \partial^+ F} \Big(Q_F(t) - Q_{F+e}(t) \Big) \mathbb{E}\big(\mu_{e,F}(t) | \boldsymbol{Q}(t), S(t) \big) \end{aligned} \quad (4.14)$$

Due to the activity constraint, if $S(t) = e$, we must have $\mu_{l,G}(t) = 0, \forall l \neq e$, for all reachable sets G. In other words, a packet can only be transmitted along the *active* edge for the mini-slot t.

For any reachable set F with an out-edge $e \in \partial^+ F$, define the weight

$$w_{F,e}(t) = Q_F(t) - Q_{F+e}(t). \quad (4.15)$$

Consider the following Max-weight policy π^*, which transmits a packet p^* belonging to class-F from node i, where the packet p^* has the highest positive weight $w_{F,e}^*(t) = \max_F w_{F,e}(t)$, from the set of all packets contending for the edge e at mini-slot t. The resulting policy is presented formally in Algorithm 6.

Algorithm 6 The Dynamic Broadcast Policy π^*

1: Select an edge e for activation independently and uniformly at random from the set of all edges E.

2: Compute all reachable sets F such that $e \in \partial^+ F$.

3: Transmit a class-F packet over the edge e, such that the corresponding weight $w_{F,e}(t) = Q_F(t) - Q_{F+e}(t)$ is positive and achieves the maximum over all such reachable sets F, computed in step 1 above. (Recall, $Q_V(t) = 0, \forall t$).

4: Idle, if no such F exists.

We now state the main theorem of this chapter.

Theorem 4.3.4 (Throughput-Optimality of π^*) *The dynamic policy π^* is a throughput-optimal broadcast policy for any network.*

Proof See Appendix (4.9.2).

Discussion Note that, the policy π^* makes dynamic routing and scheduling decision for each packet, based on the current network-state vector $\boldsymbol{Q}(t)$. In particular, its operation does not depend on the global topology information of the network. This robustness property makes it suitable for use in mobile adhoc wireless networks (MANET), where the underlying topology may change frequently.

A straightforward way to extend the resulting policy to the slotted-time model (where all edges can simultaneously transmit packets at every slot) would be to transmit a packet p_e from the class $F_e^* = \arg\max_{F:e\in\partial^+ F} w_{F,e}(t)$ over the edge $e, \forall e \in E$. Note that, the weights $w_{F,e}(t)$ are computed based on the queue-lengths $Q_F(t)$ at the beginning of slot t.

4.4 A Multi-Class Broadcasting Heuristic

A potential difficulty in implementing the policy π^* is that, one needs to maintain a state-variable $Q_F(t)$, corresponding to each reachable set F, and keep track of the particular reachable set $F_p(t)$, to which packet p belongs. For large networks, without any additional structure in the scheduling policy, maintaining such detailed state-information is quite cumbersome. To alleviate this problem, we next propose a heuristic policy which combines π^* with the idea of *in-class in-order delivery*. The introduction of class-based in-order delivery imposes additional structure in the packet scheduling, which in turn, substantially reduces the complexity of the state-space.

Motivation To motivate the heuristic policy, we begin with a simple policy-space $\Pi^{\text{in}-\text{order}}$, introduced in Chapter 2 for throughput-optimal broadcasting in wireless Directed Acyclic Graphs (DAG). Policies in $\Pi^{\text{in}-\text{order}}$ deliver packets to nodes according to their order of arrival at the source. Unfortunately, as shown in Chapter 2, although $\Pi^{\text{in}-\text{order}}$ is sufficient for achieving throughput-optimality in a DAG, it is not necessarily throughput-optimal for arbitrary networks, containing directed cycles. To tackle this problem, we generalize the idea of in-order delivery by proposing a k-class policy-space $\Pi_k^{\text{in}-\text{order}}, k \geq 1$, which generalizes the space $\Pi^{\text{in}-\text{order}}$. In this policy-space, the policies divide the packets into k distinct classes. The in-order delivery constraint is imposed within each class but not across different classes. Thus, in $\Pi_k^{\text{in}-\text{order}}$, the scheduling constraint of $\Pi^{\text{in}-\text{order}}$ is relaxed by requiring that packets belonging to *each individual class* be delivered to nodes according to their order of arrival at the source. However, the space $\Pi_k^{\text{in}-\text{order}}$ does not impose any orderly requirement for deliveries of packets across different classes. Combining it with the max-weight scheduling scheme, designed earlier for the throughput-optimal policy π^*, we propose a multi-class heuristic policy $\pi_k^H \in \Pi_k^{\text{in}-\text{order}}$ which is *conjectured* to be throughput-optimal for large-enough number of classes k. Extensive numerical simulations have been carried out to support this conjecture.

The following section gives detailed description of this heuristic policy, outlined above.

4.4.1 The In-order Policy-Space $\Pi^{\text{in-order}}$

We recall the definition of the policy-space $\Pi^{\text{in-order}}$ from Chapter 2:

Definition 4.4.1 (Policy-Space $\Pi^{\text{in-order}}$) *A broadcast policy π belongs to the space $\Pi^{\text{in-order}}$ if all incoming packets at the source $\mathtt{r}$ are serially indexed $\{1, 2, 3, \ldots\}$ according to their order of arrivals, and a node $i \in V$ is allowed to receive a packet p at time t only if the node i has received the packets $\{1, 2, \ldots, p-1\}$ by time t.*

As a result of the *in-order* delivery property of policies in the space $\Pi^{\text{in-order}}$, it follows that the state of received packets in the network at time t may be completely represented by the n-dimensional vector $\boldsymbol{R}(t)$, where $R_i(t)$ denotes the highest index of the packet received by node $i \in V$ by time t. We emphasize that this succinct representation of network-state is valid only under the action of the policies in the space $\Pi^{\text{in-order}}$, and is not necessarily true in the general policy-space Π.
Due to the highly-simplified state-space representation, it is natural to try to find efficient broadcast-policies in the space $\Pi^{\text{in-order}}$ for arbitrary network topologies. We showed in Chapter 2 that if the underlying topology of the network is restricted to DAGs, the space $\Pi^{\text{in-order}}$ indeed contains a throughput-optimal broadcast policy. However, we also proved that the space $\Pi^{\text{in-order}}$ is not rich enough to achieve broadcast capacity in networks with arbitrary topology. We re-state the following proposition in this connection.

Proposition 4.4.2 (THROUGHPUT-LIMITATION OF THE POLICY SPACE $\Pi^{\text{in-order}}$) *There exists a network $\mathcal{G}$ such that, no broadcast-policy in the space $\Pi^{\text{in-order}}$ can achieve the broadcast-capacity of $\mathcal{G}$.*

The above proposition is proved in Chapter 2, by showing that no broadcast policy in the space $\Pi^{\text{in-order}}$ can achieve the broadcast-capacity in the diamond-network $\mathcal{D}_4$, depicted in Figure 4-1.

4.4.2 The Multi-class Policy-Space $\Pi_k^{\text{in-order}}$

To overcome the throughput-limitation of the space $\Pi^{\text{in-order}}$, we propose the following generalized policy-space $\Pi_k^{\text{in-order}}, k \geq 1$, which retains the efficient representation property of the space $\Pi^{\text{in-order}}$.

Definition 4.4.3 (Policy-Space $\Pi_k^{\text{in-order}}$) *A broadcast policy π belongs to the space $\Pi_k^{\text{in-order}}$ if the following conditions hold:*

- *There are k distinct "classes".*
- *A packet, upon arrival at the source, is labelled with any one of the k classes, uniformly at random. The label of a packet remains fixed throughout its course of broadcast.*
- *Packets belonging to each individual class $j \in [1, \ldots, k]$, are serially indexed $\{1, 2, 3, \ldots\}$ according to their order of arrival.*
- *A node $i \in V$ in the network is allowed to receive a packet p from class j at time t, only if the node i has received the packets $\{1, 2, \ldots, p-1\}$ from the class j by time t .*

In other words, in the policy-space $\Pi_k^{\text{in-order}}$, packets belonging to each individual class $j \in [1, \ldots, k]$ are delivered to nodes *in-order*. It is also clear from the definition that

$$\Pi_1^{\text{in-order}} = \Pi^{\text{in-order}}$$

Thus, the collection of policy-spaces $\{\Pi_k^{\text{in-order}}, k \geq 1\}$ generalizes the policy-space $\Pi^{\text{in-order}}$.

State-Space representation under $\Pi_k^{\text{in-order}}$ Since each class in the policy-space $\Pi_k^{\text{in-order}}$ obeys the in-order delivery property, it follows that the network-state at time t is completely described by the k-tuple of vectors $\{\boldsymbol{R}^c(t), 1 \leq c \leq k\}$, where $R_i^c(t)$ denotes the highest index of the packet received by node $i \in V$ from class c by time

t. Thus the state-space complexity grows *linearly* with the number of classes used. Following our development so far, it is natural to seek a throughput-optimal broadcast policy in the space $\Pi_k^{\text{in-order}}$ with a *small* class-size k. In contrast to Proposition (4.4.2), the following proposition gives a positive result in this direction.

Proposition 4.4.4 (THROUGHPUT-OPTIMALITY OF THE SPACE $\Pi_k^{\text{in-order}}, k \geq n/2$) *For every network $\mathcal{G}$, there exists a throughput-optimal broadcast policy in the policy-space $\Pi_k^{\text{in-order}}$, for all $k \geq n/2$.*

The proof of this proposition uses a static policy, which routes the incoming packets along a set of λ^* edge-disjoint spanning trees. For a network with broadcast-capacity λ^*, the existence of these trees are guaranteed by Edmonds' tree packing theorem [14]. Then we show that for any network with unit-capacity edges, its broadcast-capacity λ^* is upper-bounded by $n/2$, which completes the proof. The details of this proof are outlined in Appendix 4.9.5.

4.4.3 General Properties of the Multi-class Policy-Space

In this subsection we show how the intra-class in-order delivery property of the multi-class policy-space constrains the delivery of packets per class. In particular, we show that at any time the number of distinct subsets of nodes, where packets from any class belong to, is at most $n + 1$. This should be contrasted with the unrestricted policy-space π, where the packets at any time may be present in all subsets of nodes, which is exponential in the size of the network.

To formally state the property, define $F_p^{(j)}(t) \subseteq V$ to be the subset of nodes where the p^{th} packet from class j belongs to at time t. We claim that,

Proposition 4.4.5 *For any $1 \leq p_1 < p_2$ and for any time t, we have*

$$F_{p_2}^{(j)}(t) \subseteq F_{p_1}^{(j)}(t) \tag{4.16}$$

Proof If $F_{p_2}^{(j)}(t) = \phi$, the inclusion holds trivially. Otherwise, consider a node $v \in F_{p_2}^{(j)}(t)$. This implies that the node v containts the ${p_2}^{\text{th}}$ packet from class k at time t. Since all classes in the policy-space $\Pi_k^{\text{in-order}}$ satisfies the in-order delivery property, it follows that the node v must contain the ${p_1}^{\text{th}}$ packet from class k at time t, where $p_1 < p_2$. Thus $v \in F_{p_1}^{(j)}(t)$. This implies that $F_{p_2}^{(j)}(t) \subseteq F_{p_1}^{(j)}(t)$, which proves the proposition.

The above proposition immediately implies the following interesting result. Let $\mathcal{F}^{(j)}(t)$ denote the family of distinct subsets of nodes where packets from class k are present at time t, i.e.,

$$\mathcal{F}^{(j)}(t) = \{F_p^{(j)}(t) | p \geq 1\} \tag{4.17}$$

Proposition 4.4.6 *For all classes $1 \leq j \leq k$ and all time $t \geq 1$, we have*

$$|\mathcal{F}^{(j)}(t)| \leq n + 1 \tag{4.18}$$

Proof Using Proposition (4.4.5), we have the following chain of set inclusions

$$V \supseteq F_1^{(j)}(t) \supseteq F_2^{(j)}(t) \supseteq \ldots \supseteq F_p^{(j)}(t) \supseteq \ldots$$

Since $|V| = n$ and the sequence of sets of vertices $\{F_i^{(j)}(t)\}_{i \geq 1}$ are *decreasing*, there could be at most $n + 1$ distinct sets in the family $\mathcal{F}^{(j)}(t)$.

Discussions Proposition 4.4.6 suggests that each individual class is structurally constrained in disseminating packets. Without the in-order restriction, we trivially have $|\mathcal{F}^{(j)}(t)| = O(2^n)$. On the other hand, under the action of any broadcast policy which routes packet along a fixed spanning tree, it is easy to see that the statement of Eqn. (4.18) holds. The surprising conclusion of Proposition 4.4.6 is that it shows that the statement of Eqn. (4.18) holds good even when we do not restrict the individual classes to follow a fixed spanning tree, but require them to respect a much weaker assumption of *in-order* delivery only. As a consequence, it is natural to search for an efficient broadcast policy with multiple classes, so that, the packet-delivery restriction of each individual class may be overcome collectively.

4.4.4 A Multi-class Heuristic Policy $\pi_k^H \in \Pi_k^{\text{in-order}}$

Since any policy in the class $\Pi_k^{\text{in-order}}$ delivers packets from the same class *in-order*, the *intra-class* packet scheduling is fixed for the entire policy-class $\Pi_k^{\text{in-order}}$. Thus, we only need to specify an *inter-class* scheduling policy to resolve contentions among multiple packets from different classes to access an active edge for transmission. In this sub-section, we propose a dynamic policy $\pi_k^H \in \Pi_k^{\text{in-order}}$, which uses the same Max-Weight packet scheduling rule, as the throughput-optimal policy π^*, for *inter-class* packet scheduling. As we will see, the computation of weights and packet scheduling in this case may be efficiently carried out by exploiting the special structure of the space $\Pi_k^{\text{in-order}}$.

Motivation We observe that, when the number of classes $k = \infty$, so that every incoming packet to the source `r` joins a new class, the *in-order* restriction of the space $\Pi_k^{\text{in-order}}$ is essentially no longer in effect. In particular, the throughput-optimal policy π^* of Section 4.3 belongs to the space $\Pi_\infty^{\text{in-order}}$. This motivates us to consider the following multi-class scheduling policy π_k^H:

Intra-class packet scheduling Recall that, under a policy $\pi \in \Pi_k^{\text{in-order}}$, a packet arriving at the source `r`, joins one of the k classes uniformly at random. Packets belonging to any class $c = 1, 2, \ldots, k$ are delivered to all nodes *in-order* (i.e. the order they arrived at the source `r`). Let the state-variable $R_i^c(t)$ denote the number of packets belonging to the class c received by node i up to the mini-slot t, $i = 1, 2, \ldots, n$, $c = 1, 2, \ldots, k$. As discussed earlier, given the intra-class in-order delivery restriction, the state of the network at the mini-slot t is completely specified by the vector $\left\{\boldsymbol{R}^c(t), c = 1, 2, \ldots, k\right\}$.

Due to the in-order packet-delivery constraint, when an edge $e = (i, j)$ is active at the mini-slot t, not all packets that are present at node i and not-present at node j are eligible for transmission. Under the policy $\pi_k^H \in \Pi_k^{\text{in-order}}$, only the next *Head-of-the-Line* (HOL) packet from each class, i.e., packet with index $R_j^c(t) + 1$ from the class c, $c = 1, 2, \ldots, k$ are eligible to be transmitted to the node j, provided that the corresponding packet is also present at node i by mini-slot t. Hence, at a given mini-slot t, there are at most k contending packets for an active edge. This should be compared with the policy π^*, in which there could be $\Theta(2^n)$ contending packets for an active edge at a mini-slot.

Inter-class packet scheduling Given the above intra-class packet-scheduling rule, which follows directly from the definition of the policy-space $\Pi_k^{\text{in-order}}$, we now propose an inter-class packet scheduling, for resolving the contention among multiple contending classes for an active edge e at a mini-slot t. For this purpose, we utilize the *same Max-Weight scheduling rule, derived for the policy* π^* (step 2 of Algorithm 6).

The main computational advantage of the multiclass policy π_k^H over the throughput-optimal policy π^* is that, instead of computing the weights $w_{F,e}(t)$ in (4.15) for all reachable sets F, we only need to compute the weights of the sets F_c corresponding to the HOL packets (if any) belonging to the class c. By exploiting the structure of the space $\Pi_k^{\text{in-order}}$, this requires quadratic number of computations in the class-size k (see Algorithm 7) per mini-slot. Finally, we schedule the HOL packet from the class

c^* having the maximum (positive) weight.

Keeping in mind our earlier discussion about similarity of packet forwarding capabilities of the classes and trees, we put forward the following conjecture regarding the performance of the proposed heuristic:

Conjecture 1 *The multiclass policy π_k^H is throughput-optimal for $k = \Theta(\lambda^*)$, where λ^* is the broadcast capacity of the network.*

Extensive numerical simulation results supporting the conjecture will be presented in Section 4.7.3.

Pseudo code The full pseudo code of the policy π_k^H is provided in Algorithm 7. In lines $4 \ldots 10$, we have used the in-order delivery property of the policy π_k^H to compute the sets F_c, to which the next HOL packet from the class c belongs. This property is also used in computing the number of packets in the set $G = F_c, F_{c+e}$ in line 14 as follows: recall that, the variable $Q_G(t)$ counts the number of packets that the reachable set G contains exclusively at mini-slot t. These packets can be counted by counting such packets from each individual classes and then summing them up. Again utilizing the in-class in-order delivery property, we conclude that the number of packets $N_G^c(t)$ from class c, that belongs exclusively to the set G at time t is given by

$$N_G^c(t) = \left(\min_{i \in G} R_i^c(t) - \max_{i \in V \setminus G} R_i^c(t) \right)^+.$$

Hence,

$$Q_G(t) = \sum_{c=1}^{k} N_G^c(t),$$

which explains the assignment in line 19. In line 23, the *weights* corresponding to the HOL packets of each class are computed according to Eqn. (4.15). Finally, in line

25, the HOL packet with the highest positive weight is transmitted across the active edge e. The per mini-slot complexity of the policy π_k^H is $\mathcal{O}(nk)$.

Algorithm 7 The Multi-class Scheduling Policy π_k^H

At each mini-slot t, the network-controller observes the state-variables $\{R_l^c(t), l \in V, c = 1, 2, \ldots, k\}$, the currently active edge $S(t) = e = (i, j)$ and executes the following steps:

for all classes $c = 1 : k$ **do**
  / Determine the index of the next expected **in-order** (HOL) packet p_c from the class c for node j */*
  $p_c \leftarrow R_j^c(t) + 1$.
  / Check whether node i has more packets than node j belonging to class c */*
  if $R_i^c(t) < p_c$ **then**
    $w_c \leftarrow 0$
    continue;
  end if
  / Find the subset $F_c \subset V$ where the packet p_c is currently present */*
  $F_c \leftarrow \phi$
  for all node $l = 1 : n$ **do**
    if $R_l^c(t) \geq p_c$ **then**
      $F_c \leftarrow F_c \cup \{l\}$
    end if
  end for
  $F_{c+e} = F_c \cup \{j\}$
  / Find $Q_{F_c}(t)$ and $Q_{F_c+e}(t)$ */*
  for $G = F_c$ and F_{c+e} **do**
    $Q_G(t) \leftarrow \sum_{c=1}^{k} \left(\min_{i \in G} R_i^c(t) - \max_{i \in V \setminus G} R_i^c(t) \right)^+$
  end for
  $Q_V(t) \leftarrow 0$
  / Compute the weight w_c for packet p_c */*
  $w_c \leftarrow \left(Q_{F_c}(t) - Q_{F_c+e}(t) \right)$
end for
Schedule the packet $p^* \in \arg\max_c w_c$, when $\max_c w_c > 0$, else idle.

4.5 Extending to Wireless Networks

A wireless network is modeled by a graph $\mathcal{G}(V, E)$, along with a set of edge-subsets $\mathcal{M}$ (represented by a set of binary characteristic vectors of dimension $|E| = m$). The set $\mathcal{M}$ is called the set of all *feasible activations* [1]. The structure of the set $\mathcal{M}$

depends on the underlying interference constraint, e.g., under the *primary interference constraint*, the set $\mathcal{M}$ consists of all *matchings* of the graph $\mathcal{G}$ [4]. Any subset of edges $\boldsymbol{s} \in \mathcal{M}$ can be activated simultaneously at a given slot. For broadcasting in wireless networks, we first activate a feasible subset of edges from $\mathcal{M}$ and then forward packets on the activated edges.

Since the proposed broadcast algorithms in sections 4.3 and 4.4 are *Max-Weight* by nature, they extend straight-forwardly to wireless networks with activation constraints [33]. In particular, from Eqn. (4.15), at each slot t, we first compute the weight of each edge, defined as $w_e(t) = \max_{F:e\in\partial^+ F} w_{e,F}(t)$. Next, we activate the subset of edges $\boldsymbol{s}^*(t)$ from the activation set $\mathcal{M}$, having the maximum weight, i.e.,

$$\boldsymbol{s}^*(t) = \arg\max_{\boldsymbol{s}\in\mathcal{M}} \sum_{e\in E} w_e(t) s_e$$

Packet forwarding over the activated edges remains the same as before. The above activation procedure carries over to the multi-class heuristic π_k^H in wireless networks.

4.6 Distributed Implementation

From the description of Algorithm 2, we note that the weight for a class c at a node i is computed based on the knowledge of the current HOL packet indices $\{R_j^c(t)\}$ of *all* nodes in the network. Gathering this global state information in a centralized fashion warrants a huge amount of control information exchanges, consuming precious network resources. To overcome this issue, we propose the following local message-passing algorithm for exchanging state information.

Each node maintains a $n \times k$ state-table consisting of the *last known* HOL packet indices $\hat{\boldsymbol{R}}(t) = \{\hat{R}_i^{(c)}(t)\}$, along with the time-stamps $t_j(t)$ of their origin. Observe that, the entry $R_i^c(t)$ corresponding to a node i is locally known to the node i, and is always fresh. However, entries corresponding to other nodes $\{\hat{\boldsymbol{R}}_j(t), j \neq i\}$ may be outdated. If an edge (i, j) is activated during a minislot, the nodes i and j exchanges their state-table w.p. q, where $0 < q \leq 1$ is a tunable parameter. The entry $\boldsymbol{R}_k(t)$

corresponding to each node k is updated at the nodes i and j with the new information available following the exchange (if any).

Let $D_{ij}(t)$ denote the age of the state-information of node j available at node i at time t. The following proposition gives an upper bound on the expectation of the maximum age of any entry at any node in the network:

Proposition 4.6.1 *Under the action of the above policy, for any connected network graph $\mathcal{G}$, we have:*

$$\mathbb{E}(\max_{i,j} D_{ij}(t)) \leq \frac{mn}{q}, \ \forall t \tag{4.19}$$

The above proposition clearly shows that the expected worst case age of state information may be reduced by increasing the parameter q, which, in turn, controls the rate of control information exchange. Proof of the above proposition is given in Appendix 4.9.6.

4.7 Numerical Simulations

4.7.1 Simulating the Throughput-optimal broadcast policy π^*

We simulate the policy π^* on the network $\mathcal{D}_4$, shown in Figure 4-1. The broadcast-capacity of the network is 2 packets per slot. External packets arrive at the source node `r` according to a Poisson process of a slightly lower rate of $\lambda = 1.95$ packets per slot. A packet is said to be broadcast when it reaches all the nodes in the network. The rate of packet arrival and packet broadcast by policy π^*, is shown in Figure 4-2. This plot exemplifies the throughput-optimality of the policy π^* in the network $\mathcal{D}_4$.

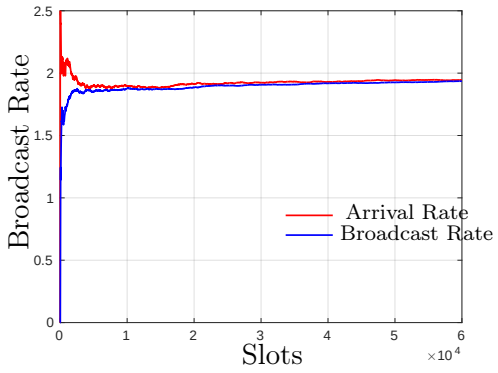


Figure 4-2: Packet Arrival and Broadcast Rate in the Diamond Network in Figure 4-1, under the action of the throughput-optimal policy π^*.

4.7.2 Simulating the Multi-class Heuristic Policy π_k^H

The multi-class heuristic policy π_k^H has been numerically simulated in 400 instances of Erdős-Rényi random network with sizes varying from $n = 20$ to $n = 40$ nodes and edge-connectivity probability $p = 0.8$. We have obtained similar qualitative results for all such instances. One representative sample is discussed here.

Consider running the broadcast-policy π_k^H on the network shown in Figure 4-3, containing $n = 20$ nodes and $m = 176$ edges. The directions of the edges in this network is chosen arbitrarily. With node 1 as the source node, we first compute the broadcast-capacity λ^* of this network using Eqn. (4.2) and obtain $\lambda^* = 6$. External packets are injected at the source node according to a Poisson process, with a slightly smaller rate of $\lambda = 0.95\lambda^* \approx 5.7$ packets per slot. The rate of broadcast under the multi-class policy π_k^H for different values of k is shown in Figure 4-4. As evident from the plot, the achievable broadcast rate, obtained by the policy π_k^H, is non-decreasing in the number of classes k. Also, the policy π_k^H broadcasts 95% of the input traffic for a relatively small value of $k = 7$.

4.7.3 Minimum Number of Classes for Achieving the Capacity

In this experiment, we simulate the heuristic multiclass policy π_k^H on two different classes of random graphs - Erdős-Rényi and Random Geometric Graphs. We randomly generate 400 instances of Erdős-Rényi graphs from the previous subsection,

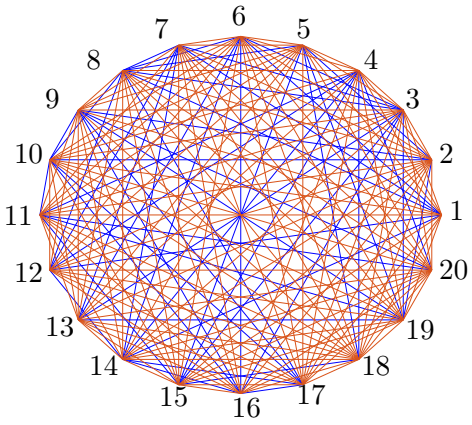


Figure 4-3: A network $\mathcal{G}$ with $N = 20$ nodes. The colors of the edges indicate their directions (e.g., *blue edge* $\implies i \to j : i > j$ and vice versa). The broadcast capacity λ^* of the network is computed to be 6, with node 1 being the source node.

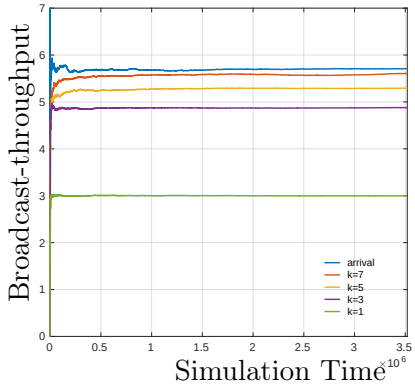


Figure 4-4: Achievable broadcast-rate with the multi-class heuristic broadcast-policies π_k^H, for $k = 1, 3, 5, 7$. The underlying network-topology is given in Figure 4-3 with broadcast capacity $\lambda^* = 6$.

along with 400 instances of two-dimensional Random Geometric Graphs with $n = 25$ nodes with varying connectivity radii [52]. For each generated graph, we first compute its broadcast capacity λ^* using Theorem 4.2.3. Packets arrive at a randomly selected node according to a Poisson process of rate 95% of the computed broadcast capacity of the graph. The empirical average of the minimum number of classes k^* required so that 95% of the incoming packets get broadcasted within $T = 2000$ slots is plotted in Figure 4-5, along with its coefficient of variation (shown by the little vertical bars). The plot is in excellent agreement with our Conjecture 1, suggesting that for a network with broadcast capacity λ^*, only $\approx \lambda^*$ classes suffice for achieving

near-broadcast-capacity, irrespective of the *size and type* of the network. Figure 4-5 also suggests that the same number of classes have different performances for two networks with different broadcast capacities.

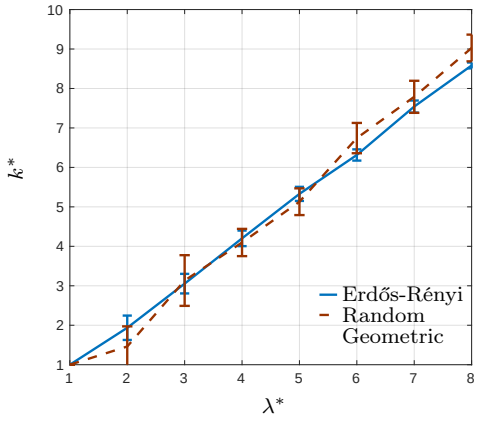


Figure 4-5: Number of classes required for achieving 95% of the broadcast capacity in Erdős-Rényi and Random Geometric Graphs.

4.8 Conclusion and Future Directions

In this chapter we studied the problem of efficient, dynamic packet broadcasting in data networks with arbitrary underlying topology. We derived a throughput-optimal Max-weight broadcast policy that achieves the capacity, albeit at the expense of using exponentially many state-variables. To get around this problem, we then proposed a multi-class heuristic policy which combines the idea of in-order packet delivery with a Max-weight scheduling, resulting in drastic reduction in complexity. The proposed heuristic with small number of classes is conjectured to be throughput-optimal. An immediate next step along this line of work would be to formally prove this conjecture. Another problem of practical interest is to find the minimum number of classes $k^*(\epsilon)$ required to achieve $(1-\epsilon)$ fraction of the capacity.

4.9 Appendix

4.9.1 Proof of Lemma (4.2.3)

Proof We prove this lemma in two parts. First, we upper-bound the achievable broadcast rate of the network under any policy in the mini-slot model by the broadcast capacity $\lambda^*(\mathcal{G})$ of the network in the usual slotted model, which is given by Eqn. (4.2). Next, in our main result in section (4.9.2), we constructively show that this rate is achievable, thus proving the lemma.

Let $\mathcal{C} \subsetneq V$ be a non-empty subset of the nodes in the graph $\mathcal{G}$ such that $\mathtt{r} \in \mathcal{C}$. Since $\mathcal{C}$ is a strict subset of V, there exists a node $i \in V$ such that $i \in \mathcal{C}^c$. Let the set $E(\mathcal{C})$ denote the set of all directed edges $e = (a, b)$ such that $a \in \mathcal{C}$ and $b \notin \mathcal{C}$. Denote $|E(\mathcal{C})|$ by $\mathrm{Cut}(\mathcal{C})$. Using the Max-Flow-Min-Cut theorem [49], the broadcast-capacity in the slotted model, given by Eqn. (4.2), may be alternatively represented as

$$\lambda^* = \min_{\mathcal{C} \subsetneq V, \mathtt{r} \in \mathcal{C}} \mathrm{Cut}(\mathcal{C}) \tag{4.20}$$

Now let us proceed with the mini-slot model. Since all packets arrived at source $\mathtt{r}$ that are received by the node i must cross some edge in the cut $E(\mathcal{C})$, it follows that, under any policy $\pi \in \Pi$, the total number of packets $R_i(t)$ that are received by node i up to mini-slot t is upper-bounded by

$$R_i(t) \leq \sum_{\tau=1}^{t} \sum_{e \in E(\mathcal{C})} \mathbb{1}(S(\tau) = e) = \sum_{e \in E(\mathcal{C})} \sum_{\tau=1}^{t} \mathbb{1}(S(\tau) = e) \tag{4.21}$$

Thus the broadcast-rate $\lambda^{\pi}_{\text{mini-slot}}$ achievable in the mini-slot model is upper-bounded by

$$\lambda^{\pi}_{\text{mini-slot}} \overset{(a)}{\leq} \liminf_{t\to\infty} \frac{R_i(t)}{t} \tag{4.22}$$

$$\overset{(b)}{\leq} \liminf_{t\to\infty} \frac{1}{t} \sum_{e\in E(\mathcal{C})} \sum_{\tau=1}^{t} \mathbb{1}(S(\tau)=e)$$

$$= \sum_{e\in E(\mathcal{C})} \lim_{t\to\infty} \frac{1}{t} \sum_{\tau=1}^{t} \mathbb{1}(S(\tau)=e) \tag{4.23}$$

$$\overset{(c)}{=} \frac{1}{m}\text{Cut}(\mathcal{C}),\ \text{w.p.1} \tag{4.24}$$

Where the inequality (a) follows from the definition of broadcast-rate (4.1), inequality (b) follows from Eqn. (4.21) and finally, the equality (c) follows from the Strong Law of Large Numbers [50]. Since the inequality (6.7) holds for any cut $\mathcal{C} \subsetneq C$ containing the source `r` and any policy π, from Eqn. (4.20) we have

$$\lambda^{*}_{\text{mini-slot}} \leq \lambda^{\pi}_{\text{mini-slot}} \leq \frac{1}{m}\text{Cut}(\mathcal{C}) \leq \frac{1}{m}\lambda^{*} \text{ per mini-slot} \tag{4.25}$$

Since according to the hypothesis of the lemma, a slot is identified with m mini-slots, the above result shows that

$$\lambda^{*}_{\text{mini-slot}} \leq \lambda^{*} \text{ per slot} \tag{4.26}$$

This proves that the capacity in the mini-slot model (per slot) is at most the capacity of the slotted-time model (given by Eqn. (4.2)). In section (4.3), we show that there exists a broadcast policy $\pi^{*} \in \Pi$ which achieves a broadcast-rate of λ^{*} packets per-slot in the mini-slot model. This concludes the proof of the lemma.

4.9.2 Proof of Throughput Optimality of π^{*}

In this subsection, we show that the induced Markov-Chain $\boldsymbol{Q}^{\pi^{*}}(t)$, generated by the policy π^{*} is positive recurrent, for all arrival rates $\lambda < \lambda^{*}$ packets per slot. This

is proved by showing that the expected one-minislot drift of the Lyapunov function $L(\boldsymbol{Q}(t))$ is negative outside a bounded region in the non-negative orthant $\mathbb{Z}_+^M$, where M is the dimension of the state-space $\boldsymbol{Q}(t)$. To establish the required drift-condition, we first construct an auxiliary stationary randomized policy π^{RAND}, which is easier to analyze. Then we bound the one-minislot expected drift of the policy π^* by comparing it with the policy π^{RAND}.

We emphasize that the construction of the randomized policy π^{RAND} is highly non-trivial, because under the action of the policy π^*, a packet may travel along an arbitrary tree and as a result, *any* reachable set $F \in \mathcal{F}$ may potentially contain non-zero number of packets.

For ease of exposition, the proof of throughput-optimality of the policy π^* is divided into several parts.

Part I: Consequence of Edmonds' Tree-packing Theorem

From Edmond's tree-packing theorem [14], it follows that the graph $\mathcal{G}$ contains λ^* edge-disjoint directed spanning trees, [6] $\{\mathcal{T}^i\}_1^{\lambda^*}$. From Proposition (4.2.3) and Lemma (4.3.3), it follows that, to prove the throughput-optimality of the policy π^*, it is sufficient to show stochastic-stability of the process $\{\boldsymbol{Q}(t)\}_0^\infty$ for an arrival rate of λ/m per minislot, where $\lambda < \lambda^*$.

Fix an arbitrarily small $\epsilon > 0$ such that,

$$\lambda \leq \lambda^* - \epsilon$$

Now we construct a stationary randomized policy π^{RAND}, which utilizes the edge-disjoint trees $\{\mathcal{T}^i\}_{i=1}^{\lambda^*}$ in a critical fashion.

[6] Note that, since the edges are assumed to be of unit capacity, λ^* is an integer. This result follows by combining Eqn. (4.2) with the Max-Flow-Min-Cut theorem [49].

Part II: Construction of a Stationary Randomized Policy $\pi^{\mathbf{RAND}}$:

The stationary randomized policy π^{RAND} allocates rates $\mu_{e,F}(t)$ randomly to different ordered pairs (e, F), for transmitting packets belonging to reachable sets F, across an edge $e \in \partial^+ F$ [7]. Recall that $\mu_{e,F}(t)$'s are binary variables. Hence, conditioned on the edge-activity process $S(t) = e$, the allocated rates are fully specified by the set of probabilities that a packet from the reachable set F is transmitted across the active edge $e \in \partial^+ F$. Equivalently, we may specify the allocated rates in terms of their expectation w.r.t. the edge-activation process (obtained by multiplying the corresponding probabilities by $1/m$).

Informally, the policy π^{RAND} allocates most of the rates along the reachable sequences corresponding to the edge-disjont spanning trees $\{\mathcal{T}^i\}_1^{\lambda^*}$, obtained in Part I. However, since the dynamic policy π^* is not restricted to route packets along the spanning trees $\{\mathcal{T}^i\}_1^{\lambda^*}$ only, for technical reasons which will be evident later, π^{RAND} is designed to allocate small amount of rates along other reachable sequences. This is an essential and non-trivial part of the proof methodology. An illustrative example of the rate allocation strategy by the policy π^{RAND} will be described subsequently for the diamond graph $\mathcal{D}_4$ of Figure 4-1.

Formally, the rate-allocation by the randomized policy π^{RAND} is given as follows:

- We index the set of all reachable sequences in a specific order.
  - The first λ^* reachable sequences $\{\zeta^i\}_{i=1}^{\lambda^*}$ are defined as follows: for each edge-disjoint tree $\mathcal{T}^i, i = 1, 2, \ldots, \lambda^*$ obtained from Part-I, recursively construct a reachable sequence $\zeta^i = \{(F_j^i, e_j^i)\}_{j=1}^{n-1}$, such that the induced subgraphs $\mathcal{T}^i(F_j^i)$ are connected for all $j = 1, 2, \ldots, n-1$.
    In other words, for all $1 \leq i \leq \lambda^*$ define $F_1^i = \{\mathtt{r}\}$ and for all $1 \leq j \leq n-2$, the set F_{j+1}^i is recursively constructed from the set F_j^i by adding a node to the set F_j^i while traversing along an edge of the tree $\mathcal{T}^i$. Let the corresponding edge in $\mathcal{T}^i$ connecting the $j+1$ $^{\text{th}}$ vertex $F_{j+1}^i \setminus F_j^i$, to the set F_j^i, be e_j^i. Since the trees $\{\mathcal{T}^i\}_{i=1}^{\lambda^*}$ are edge disjoint, the edges e_j^i's are **distinct**

[7] If $e \notin \partial^+ F$, naturally $\mu_{e,F}(t) = 0, \forall t$.

for all $i = 1, 2, \ldots, \lambda^*$ and $j = 1, 2, \ldots, n-1$. The above construction defines the first λ^* reachable sequences $\zeta^i = \{F^i_j, e^i_j\}_{j=1}^{n-1}, 1 \leq i \leq \lambda^*$.

– In addition to the above, let $\{\zeta^i = (F^i_j, e^i_j)\}_{j=1}^{n-1}, \lambda^*+1 \leq i \leq B$ be the set of all *other* reachable sequence in the graph $\mathcal{G}$, different from the previously constructed λ^* reachable sequences. Recall that, B is the cardinality of the set of all reachable sequences in the graph $\mathcal{G}$. Thus the set of *all* reachable sequences in the graph $\mathcal{G}$ is given by $\bigcup_{i=1}^{B} \zeta^i$.

- To define the expected allocated rates $\mathbb{E}\mu_{e,F}(t)$, it is useful to first define some auxiliary variables, called *rate-components* $\mathbb{E}\mu^i_{e,F}(t), i = 1, 2, \ldots, B$, corresponding to each reachable sequence. The rate $\mathbb{E}\mu_{e,F}(t)$ is is simply the sum of the rate-components, as given in Eqn. (4.29).
  At each slot t and $1 \leq i \leq \lambda^*$, the randomized policy allocates i^{th} *rate-component* corresponding to the reachable sequence $\zeta^i = \{e^i_j, F^i_j\}_{j=1}^{n-1}$ according to the following scheme:

$$\begin{aligned} \mathbb{E}\big(\mu^i_{e^i_j, F^i_j}(t)\big) &= 1/m - \epsilon(n-j)/n, \\ &\quad \forall\, 1 \leq j \leq n-1 \\ &= 0, \quad \text{o.w.} \end{aligned} \tag{4.27}$$

- In addition to the rate-allocation (4.27), the randomized policy π^{RAND} also allocates small amount of rates corresponding to other reachable sequences $\{\zeta^i\}_{\lambda^*+1}^{B}$ according to the following scheme: For $\lambda^* + 1 \leq i \leq B$, the randomized policy allocates i^{th} rate-component to the ordered pairs (e, F) as follows:

$$\begin{aligned} \mathbb{E}\big(\mu^i_{e^i_j, F^i_j}(t)\big) &= \frac{\epsilon}{2nB} - \frac{\epsilon}{2nB}\frac{n-j}{n}, \\ &\quad \forall\, 1 \leq j \leq n-1, \\ &= 0, \quad \text{o.w.} \end{aligned} \tag{4.28}$$

The overall rate allocated to the pair (e, F) is simply the sum of the component-rates, as given below:

$$\mathbb{E}\mu_{e,F}(t) = \sum_{i=1}^{B} \mathbb{E}\mu^{i}_{e,F}(t) \tag{4.29}$$

In the following, we show that the above rate-allocation is feasible with respect to the edge capacity constraint.

Lemma 4.9.1 (Feasibility of Rate Allocation) *The rate allocation* (4.29) *by the randomized policy* π^{RAND} *is feasible.*

The reader is referred to Appendix (4.9.3) for the proof the lemma. An illustrative example for the above randomized rate-allocation scheme is given in Appendix (4.9.4).

Part III: Comparison of drifts under action of policies π^* and π^{RAND}

Recall that, from Eqn. (4.14) we have the following upper-bound on the one-minislot drift of the Lyapunov function $L(\boldsymbol{Q}(t)$, achieved by the policy π^*:

$$(\Delta^{\pi^*}(\boldsymbol{Q}(t)|S(t)) \leq 2^n \mu^2_{\max} - \sum_{(e,F):e\in\partial^+ F} \Big(Q_F(t) - Q_{F+e}(t)\Big)\mathbb{E}\big(\mu^{\pi^*}_{e,F}(t)|\boldsymbol{Q}(t), S(t)\big)$$

Since the policy π^*, by definition, transmits packets to maximize the weight $w_{F,e}(t) = Q_F(t) - Q_{F+e}(t)$ point wise, the following inequality holds

$$\sum_{(e,F):e\in\partial^+ F} \Big(Q_F(t) - Q_{F+e}(t)\Big)\mathbb{E}\big(\mu^{\pi^*}_{e,F}(t)|\boldsymbol{Q}(t), S(t)\big) \geq$$
$$\sum_{(e,F):e\in\partial^+ F} \Big(Q_F(t) - Q_{F+e}(t)\Big)\mathbb{E}\big(\mu^{\pi^{\text{RAND}}}_{e,F}(t)|\boldsymbol{Q}(t), S(t)\big),$$

where the randomized rate-allocation $\boldsymbol{\mu}^{\pi^{\text{RAND}}}$ is given by Eqn. (4.29). Noting that π^{RAND} operates independently of the "queue-states" $\boldsymbol{Q}(t)$ and dropping the super-script π^{RAND} from the control variables $\boldsymbol{\mu}(t)$ on the right hand side, we can bound

the one-slot expected drift of the policy π^* as follows:

$$\begin{aligned}
& (\Delta^{\pi^*}(\boldsymbol{Q}(t))|S(t)) \\
\leq\ & 2^n\mu_{\max}^2 - \sum_{(e,F):e\in\partial^+F}\Big(Q_F(t) - Q_{F+e}(t)\Big)\mathbb{E}\big(\mu_{e,F}(t)|S(t)\big) \\
=\ & 2^n\mu_{\max}^2 - \sum_F Q_F(t)\Big(\sum_{e\in\partial^+F}\mathbb{E}(\mu_{e,F}(t)|S(t)) - \sum_{(e,G):e\in\partial^-F,G=F\setminus\{e\}}\mathbb{E}(\mu_{e,G}(t)|S(t))\Big) \\
\overset{(a)}{=}\ & 2^n\mu_{\max}^2 - \sum_F Q_F(t)\Big(\sum_{e\in\partial^+F}\big(\sum_{i=1}^B\mathbb{E}(\mu_{e,F}^i(t)|S(t))\big) - \sum_{(e,G):e\in\partial^-F,G=F\setminus\{e\}}\big(\sum_{i=1}^B\mathbb{E}(\mu_{e,G}^i(t)|S(t))\big)\Big),
\end{aligned}$$

where in (a) we have used Eqn. (4.29).

Taking expectation of both sides of the above inequality w.r.t the random edge-activation process $S(t)$ and interchanging the order of summation, we have

$$\Delta^{\pi^*}(\boldsymbol{Q}(t)) \leq 2^n\mu_{\max}^2 - \sum_F Q_F(t)\sum_{i=1}^B\Big(\sum_{e\in\partial^+F}\mathbb{E}(\mu_{e,F}^i(t)) - \sum_{(e,G):e\in\partial^-F,G=F\setminus\{e\}}\mathbb{E}(\mu_{e,G}^i(t))\Big), \tag{4.30}$$

where the rate-components $\boldsymbol{\mu}^i$ of the randomized policy π^{RAND} are defined in Eqns (4.27) and (4.28).

Fix a reachable set F, appearing in the outer-most summation of the above upper-bound (4.30). Now focus on the i^{th} reachable sequence $\zeta^i \equiv \{F_j^i, e_j^i\}_1^{n-1}$. We have two cases:

Case I: $F \notin \zeta^i$

Here, according to the allocations in (4.27) and (4.28), we have

$$\sum_{e\in\partial^+F}\mathbb{E}(\mu_{e,F}^i(t)) \overset{(a)}{=} 0, \quad \sum_{(e,G):e\in\partial^-F,G=F\setminus\{e\}}\mathbb{E}(\mu_{e,G}^i(t)) \overset{(b)}{=} 0$$

Where the equality (a) follows from the assumption that $F \notin \zeta^i$ and equality (b) follows from the fact that positive rates are allocated only along the tree corresponding to the reachable sequence ζ^i. Hence, if no rate is allocated to drain packets outside the set F, π^{RAND} does not allocate any rate to route packets to the set F.

Case II: $F \in \zeta^i$

In this case, from Eqns. (4.27) and (4.28), it follows that

$$\Bigg(\sum_{e \in \partial^+ F} \mathbb{E}(\mu^i_{e,F}(t)) - \sum_{(e,G): e \in \partial^- F, G = F \setminus \{e\}} \mathbb{E}(\mu^i_{e,G}(t)) \Bigg) = \begin{cases} \frac{\epsilon}{n}, & 1 \leq i \leq \lambda^* \\ \frac{\epsilon}{2n^2 B}, & \lambda^* + 1 \leq i \leq B \end{cases} \tag{4.31}$$

By definition, each reachable set is visited by at least one reachable sequence. In other words, there exists at least one $i, 1 \leq i \leq B$, such that $F \in \zeta^i$. Combining the above two cases, from the upper-bound (4.30) we conclude that

$$\Delta^{\pi^*}(\boldsymbol{Q}(t)) \leq 2^n \mu^2_{\max} - \frac{\epsilon}{2n^2 B} \sum_F Q_F(t), \tag{4.32}$$

where, the sum extends over *all* reachable sets. The drift is negative, i.e., $\Delta^{\pi^*}(\boldsymbol{Q}(t)) < -\epsilon$, when $\boldsymbol{Q}_F \in \mathcal{B}^c$, where

$$\mathcal{B} = \left\{ (Q_F \geq 0) : \sum_F Q_F \geq \frac{2n^2 B}{\epsilon} (\epsilon + 2^n \mu^2_{\max}) \right\}$$

Invoking the Foster-Lyapunov criterion [51], we conclude that the Markov-Chain $\{\boldsymbol{Q}^{\pi^*}(t)\}_0^\infty$ is positive recurrent. Finally, throughput-optimality of the policy π^* follows from lemma 4.3.3. ■

4.9.3 Proof of Lemma (4.9.1)

Proof The rate allocation (4.29) will be feasible if the sum of the allocated probabilities that an active edge e carries a class-F packet, for all reachable sets F, is at most unity. Since an edge can carry at most one packet per mini-slot, this feasibility condition is equivalent to the requirement that the total expected rate, i.e., $\mathbb{E}\mu_e(t) = \sum_F \mathbb{E}\mu_{e,F}(t)$, allocated to an edge $e \in E$ by the randomized policy π^{RAND} does not exceed $\frac{1}{m}$ (the expected capacity of the edge per mini-slot).

Since an edge e may appear at most once in any reachable sequence, the total rate allocated to an edge e by the randomized-policy π^{RAND} is upper-bounded by $\frac{1}{m} - \frac{\epsilon}{n} + (B - \lambda^*)\frac{\epsilon}{2nB} \leq \frac{1}{m} - \frac{\epsilon}{2n} < 1/m$. Hence the rate allocation by the randomized policy π^{RAND} is feasible.

4.9.4 An Example of Rate Allocation by the Stationary policy π^{RAND}

As an explicit example of the above stationary policy, consider the case of the Diamond network $\mathcal{D}_4$, shown in Figure 4-1. The edges of the trees $\{\mathcal{T}^i, i = 1, 2\}$ are shown in blue and red colors in the figure. Then the randomized policy allocates the following rate-components to the edges, where the expectation is taken w.r.t. random edge-activations per mini-slot.

First we construct a reachable sequence ζ^1 consistent with the tree $\mathcal{T}^1$ as follows:

$$\zeta^1 = \{(\{\mathtt{r}\}, \mathtt{ra}), (\{\mathtt{r,a}\}, \mathtt{ab}), (\{\mathtt{r,a,b}\}, \mathtt{bc})\}$$

Next we allocate the following rate-components as prescribed by π^{RAND}:

$$\begin{aligned}
\mathbb{E}\mu^1_{\mathtt{ra},\{\mathtt{r}\}}(t) &= 1/6 - 3\epsilon/4 \\
\mathbb{E}\mu^1_{\mathtt{ab},\{\mathtt{r,a}\}}(t) &= 1/6 - 2\epsilon/4 \\
\mathbb{E}\mu^1_{\mathtt{bc},\{\mathtt{r,a,b}\}}(t) &= 1/6 - \epsilon/4 \\
\mathbb{E}\mu^1_{e,F}(t) &= 0, \quad \text{o.w.}
\end{aligned}$$

Similarly for the tree $\mathcal{T}^2$, we first construct a reachable sequence ζ^2 as follows:

$$\zeta^2 = \{(\{\mathtt{r}\}, \mathtt{rb}), (\{\mathtt{r,b}\}, \mathtt{rc}), (\{\mathtt{r,b,c}\}, \mathtt{ca})\}$$

Then we allocate the following component-rates to the (edge, set) pairs as follows:

$$\begin{aligned}
\mathbb{E}\mu^2_{\mathtt{rb},\{\mathtt{r}\}}(t) &= 1/6 - 3\epsilon/4 \\
\mathbb{E}\mu^2_{\mathtt{rc},\{\mathtt{r,b}\}}(t) &= 1/6 - 2\epsilon/4 \\
\mathbb{E}\mu^2_{\mathtt{ca},\{\mathtt{r,b,c}\}}(t) &= 1/6 - \epsilon/4 \\
\mathbb{E}\mu^2_{e,F}(t) &= 0, \quad \text{o.w.}
\end{aligned}$$

In this example $\lambda^* = 2$, thus these two reachable sequence accounts for a major portion of the rates allocated to the edges. The randomized policy π^{RAND}, however, allocates small rates to other reachable sequences too. As an example, consider the following reachable sequence ζ^3, given by

$$\zeta^3 = \{(\{\mathtt{r}\}, \mathtt{ra}), (\{\mathtt{r,a}\}, \mathtt{rb}), (\{\mathtt{r,a,b}\}, \mathtt{rc}\}$$

Then, as prescribed above, the randomized policy allocates the following rate-components

$$\begin{aligned}
\mathbb{E}\mu^3_{\mathtt{ra},\{\mathtt{r}\}}(t) &= \frac{\epsilon}{8B} - \frac{3\epsilon}{32B} \\
\mathbb{E}\mu^3_{\mathtt{rb},\{\mathtt{r,a}\}}(t) &= \frac{\epsilon}{8B} - \frac{2\epsilon}{32B} \\
\mathbb{E}\mu^3_{\mathtt{rc},\{\mathtt{r,a,b}\}}(t) &= \frac{\epsilon}{8B} - \frac{\epsilon}{32B} \\
\mathbb{E}\mu^3_{e,F}(t) &= 0, \quad \text{o.w.}
\end{aligned}$$

Here B is the number of all distinct reachable sequences, which is upper-bounded by 4^8. The rate-components corresponding to other reachable sequences may be computed as above. Finally, the actual expected rate-allocation to the pair (e, F) is given by

$$\mathbb{E}\mu_{e,F}(t) = \sum_{i=1}^{B} \mathbb{E}\mu^i_{e,F}(t)$$

4.9.5 Proof of Proposition (4.4.4)

The proof of this proposition is conceptually simplest in the slotted-time model. The argument also applies directly to the mini-slot model.

Consider a network $\mathcal{G}$ with broadcast-capacity λ^*. Assume a slotted-time model. By Edmonds' tree-packing Theorem [14], we know that there exists λ^* number of edge-disjoint directed spanning trees (arborescences) $\{\mathcal{T}_i\}_1^{\lambda^*}$ in $\mathcal{G}$, rooted at the source node `r`. Now consider a policy $\pi \in \Pi_k^{\text{in-order}}$ with $k \geq \lambda^*$ which operates as follows:

- An incoming packet is placed in any of the classes $[1, 2, 3, \ldots, \lambda^*]$, uniformly at random.
- Packets in a class i are routed to all nodes in the network *in-order* along the directed tree $\mathcal{T}_i$, where the packets are replicated in all non-leaf nodes of the tree $\mathcal{T}_i, 1 \leq i \leq \lambda^*$.

Since the trees are edge-disjoint, the classes do not conflict; i.e., routing in each class can be carried out independently. Also by the property of $\mathcal{T}_i$, there is a *unique* directed path from the source node `r` to any other node in the network along the edges of the tree $\mathcal{T}_i, 1 \leq i \leq \lambda^*$. Thus packets in every class can be delivered to all nodes in the network *in-order* in a pipe-lined fashion with the long-term delivery-rate of 1 packet per class. Since there are λ^* packet-carrying classes, it follows that the policy $\pi \in \Pi_k^{\text{in-order}}$ is throughput-optimal for $k \geq \lambda^*$.

Next we show that, $\lambda^* \leq n/2$ for a simple network. Since there exist λ^* number of edge-disjoint directed spanning trees in the network, and since each spanning-tree contains $n-1$ edges, we have

$$\lambda^*(n-1) \leq m \tag{4.33}$$

Where m is the number of edges in the network. But we have $m \leq n(n-1)/2$ for a simple graph. Thus, from the above equation, we conclude that

$$\lambda^* \leq n/2. \tag{4.34}$$

This completes the proof of the Proposition.

4.9.6 Proof of Proposition (4.6.1)

Proof Consider a spanning tree $\mathcal{T}$ in the network (it exists, as the network is assumed to be connected). One of the many possible ways to send its state information from any node j to a node i would be to send this information following the unique path $\mathcal{P}_{ij}$ induced by the tree $\mathcal{T}$. Thus,

$$D_{ij}(t) \leq \sum_{e \in \mathcal{P}_{ij}} X_e, \tag{4.35}$$

where X_e is the (stationary) random variable denoting the number of minislots one has to wait until a state exchange takes place along the edge e. Hence, it follows that

$$\max_{i,j} D_{ij}(t) \leq \max_{i,j} \sum_{e \in \mathcal{P}_{ij}} X_e \overset{(a)}{\leq} \sum_{e \in \mathcal{T}} X_e. \tag{4.36}$$

Where the inequality (a) follows from the fact that the r.v.s X_e's are non-negative. Since X_e's are geometrically distributed with parameter $\frac{q}{m}$, we have $\mathbb{E}X_e = \frac{m}{q}$. Taking expectation of both sides of (4.36), we have

$$\mathbb{E}\big(\max_{i,j} D_{ij}(t)\big) \overset{(a)}{\leq} \sum_{e \in \mathcal{T}} \mathbb{E}X_e \leq \frac{mn}{q},$$

where the inequality (a) follows from the fact that there are exactly $n-1$ edges in the spanning tree $\mathcal{T}$.

Chapter 5

Optimal Control for Generalized Network-Flow Problems

5.1 Overview of the Results

In this chapter we consider the **Generalized Flow Problem** (**GNF**), by proposing a new throughput-optimal routing and scheduling policy, called **Universal Max-Weight** (**UMW**), for an arbitrary wireless network carrying diverse flows, including but not limited to **Unicast, Broadcast, Multicast and Anycast** traffic. Hence, unlike the previous chapters which focusses on the Broadcast problem exclusively, this chapter provides a unified framework for solving *all* network flow problems in general. Shortly, we will see that, **UMW** provides a very different policy than the well-known Backpressure policy, when specialized to the Unicast setting.

The proposed **UMW** policy uses a *virtual network of queues* - one virtual queue per link in the network. We solve the routing problem dynamically using a simple "weighted-shortest-route" computation on the virtual network and using the corresponding route on the physical network. Optimal link scheduling is performed by a max-weight computation, also in the virtual network, and then using the resulting activation in the physical network. The overall algorithm is dynamic, cycle-free, and solves the generalized routing and scheduling problem optimally (i.e., maximally stable or throughput optimal). In addition to this, the proposed **UMW** policy has the

following advantages:

1. **Generalized Solution:** Unlike the BP policy, which solves only the unicast problem, the proposed **UMW** policy efficiently addresses all of the aforementioned network flow problems in both wired and wireless networks in a very general setting.

2. **Delay Reduction:** Although the celebrated BP policy is throughput-optimal, its average delay performance is known to be poor due to the occurrence of packet-cycling in the network [25] [53]. In our proposed **UMW** policy, each packet traverses a dynamically selected *acyclic* route, which drastically reduces the average latency.

3. **State-Complexity Reduction:** Unlike the BP policy, which maintains *per-flow* queues at each node, the proposed **UMW** policy maintains only a virtual-queue counter and a priority queue per link, irrespective of the number and type of flows in the network. This reduces the amount of overhead that needs to be maintained for efficient operation.

4. **Efficient Implementation:** In the BP policy, routing decisions are made hop-by-hop by the intermediate nodes. This puts a considerable amount of computational overhead on the individual nodes. In contrast, in the proposed **UMW** policy, the entire route of the packets is determined at the source (similar to the *dynamic source routing* [54]). Hence, the entire computational requirement is transferred to the source, which often has higher computational/energy resources than the nodes in the rest of the network (*e.g.*, wireless sensor networks).

The rest of the chapter is organized as follows: In section 5.2 we discuss the basic system model and formulate the problem. In section 5.3 we give a brief overview of the proposed **UMW** policy. In section 5.4 we discuss the structure and dynamics of the virtual queues, on which **UMW** is based. In section 5.5 we prove its stability property in the multi-hop physical network. In section 5.6 we discuss implementation details. In section 5.7 we provide extensive simulation results, comparing **UMW**

with other competing algorithms. In section 5.8 we conclude the chapter with a few directions for further research.

5.2 Network-Layer Capacity Region

In this section, we characterize the network-layer capacity region in the presence of generalized flows.

5.2.1 Admissible Routes of Packets

Without any loss of throughput-optimality, we can concentrate on policies which delivers any packet p to any node in the network *at most* once.[1] This immediately implies that the set of all admissible routes $\mathcal{T}^{(c)}$ for packets of any class c, in general, comprises of trees rooted at the corresponding source node $s^{(c)}$. In particular, depending on the type of class c traffic, the topology of the admissible routes $\mathcal{T}^{(c)}$ takes the following special forms:

- **Unicast Traffic**: $\mathcal{T}^{(c)}$ = set of all $s^{(c)} - t^{(c)}$ paths in the graph $\mathcal{G}$.
- **Broadcast Traffic**: $\mathcal{T}^{(c)}$ = set of all spanning trees in the graph $\mathcal{G}$, rooted at $s^{(c)}$.
- **Multicast Traffic**: $\mathcal{T}^{(c)}$ = set of all Steiner trees [22] in $\mathcal{G}$, rooted at $s^{(c)}$ and spanning the vertices $\mathcal{D}^{(c)} = \{t_1^{(c)}, t_2^{(c)}, \ldots, t_k^{(c)}\}$.
- **Anycast Traffic**: $\mathcal{T}^{(c)}$ = union of all $s^{(c)} - t_i^{(c)}$ paths in the graph $\mathcal{G}$, $i = 1, 2, \ldots, k$.

[1] This should be contrasted with the popular throughput-optimal unicast policy *Back-Pressure* [1], which does not satisfy this constraint and may deliver the same packet to a node multiple times, thus potentially degrading its delay performance.

5.2.2 Characterization of the Network-Layer Capacity Region

Consider any arrival vector $\boldsymbol{\lambda} \in \boldsymbol{\Lambda}(\mathcal{G}, \mathcal{C})$. By definition, there exists an admissible policy $\pi \in \Pi$, which supports the arrival rate $\boldsymbol{\lambda}$ by means of storing, duplicating and forwarding packets efficiently. Taking time-averages over the actions of the policy π, it is clear that there exist a *randomized* flow-decomposition and scheduling policy to route the packets such that none of the edges in the network is overloaded. Indeed, in the following theorem, we show that for every $\boldsymbol{\lambda} \in \boldsymbol{\Lambda}(\mathcal{G}, \mathcal{C})$, there exist non-negative scalars $\{\lambda_i^{(c)}\}$, indexed by the admissible routes $T_i^{(c)} \in \mathcal{T}^{(c)}$ and a convex combination of the link activation vectors $\overline{\boldsymbol{\mu}} \in \text{conv}(\mathcal{M})$ such that,

$$\lambda^{(c)} = \sum_{T_i^{(c)} \in \mathcal{T}^{(c)}} \lambda_i^{(c)}, \quad \forall c \in \mathcal{C} \tag{5.1}$$

$$\lambda_e \overset{\text{(def.)}}{=} \sum_{(i,c): e \in T_i^{(c)}, T_i^{(c)} \in \mathcal{T}^{(c)}} \lambda_i^{(c)} \leq \overline{\mu}_e, \quad \forall e \in E. \tag{5.2}$$

Eqn. (5.1) denotes decomposition of the average incoming flows into different admissible routes and Eqn. (5.2) denotes the fact that none of the edges in the network is overloaded, i.e. arrival rate of packets to any edge e under the policy π is *at most* the rate allocated by the policy π to the edge e to serve packets.

To state the result precisely, define the set $\overline{\boldsymbol{\Lambda}}$ to be the set of all arrival vectors $\boldsymbol{\lambda} \in \mathbb{R}_+^{|\mathcal{C}|}$, for which there exists a randomized activation vector $\boldsymbol{\mu} \in \text{conv}(\mathcal{M})$ and a non-negative flow decomposition $\{\lambda_i^{(c)}\}$, such that Eqns. (5.1) and (5.2) are satisfied. We have the following theorem:

Theorem 5.2.1 *The network-layer capacity region $\boldsymbol{\Lambda}(\mathcal{G}, \mathcal{C})$ is characterized by the set $\overline{\boldsymbol{\Lambda}}$, up to its boundary.*

Proof of Theorem 5.2.1 consists of two parts: converse and achievability. Proof of the *converse* is given in Appendix 5.9.1, where we show that all supportable arrival rates must belong to the set $\overline{\boldsymbol{\Lambda}}$. The main result of this chapter, as developed in

the subsequent sections, is the construction of an efficient admissible policy, called **U**niversal **M**ax-**W**eight (**UMW**), which *achieves* any arrival rate in the interior of the set $\overline{\mathbf{\Lambda}}$.

5.3 Overview of the UMW Policy

In this section, we present a brief overview of our throughput-optimal **UMW** policy, designed and analyzed in the subsequent sections. Central to the **UMW** policy is a global state vector, called virtual queues $\tilde{\boldsymbol{Q}}(t)$, used for packet routing and link activations. Each component of the virtual queues is updated at every slot according to a one-hop queueing (Lindley) recursion, corresponding to a *relaxed* network, described in detail in section 5.4. Unlike the well-known Back-Pressure algorithm for the unicast problem [1], in which packet routing decisions are made hop-by-hop using physical queue lengths $\boldsymbol{Q}(t)$, the **UMW** policy prescribes an admissible route to each incoming packet immediately upon its arrival (dynamic source routing). This route selection decision is dynamically made by solving a suitable min-cost routing problem (e.g., shortest path, MST etc.) at the source with edge costs given by the current virtual-queue vector $\tilde{\boldsymbol{Q}}(t)$. Link activation decisions at each slot are made by a Max-Weight algorithm with link-weights set equal to $\tilde{\boldsymbol{Q}}(t)$. Having fixed the routing and activation policy as above, in section 5.5 we design a packet scheduling algorithm for the physical network, which efficiently resolves contention among multiple packets that wait to cross the same (active) edge at the same slot. We show that the overall policy is throughput-optimal. One significantly new feature of our algorithm is that it is entirely oblivious to the length of the physical queues of the network and utilizes the auxiliary virtual-queue state variables for stabilizing the former.

Our proof of throughput-optimality of **UMW** leverages ideas from *deterministic* adversarial queueing theory and combines it effectively with the *stochastic* Lyapunov-drift based techniques and may be of independent theoretical interest.

5.4 Global Virtual Queues: Structures, Algorithms, and Stability

In this section, we introduce the notion of *virtual queues* [2], which is obtained by *relaxing* the dynamics of the physical queues of the network in the following intuitive fashion.

5.4.1 Precedence Constraints

In a multi-hop network, if a packet p is being routed along the path $T = l_1 - l_2 - \ldots - l_k$, where $l_i \in E$ is the i^{th} link on its path, then by the principle of causality, the packet p cannot be physically transmitted over the j^{th} link l_j if it has not already been transmitted by the first $j-1$ links $l_1, l_2, \ldots, l_{j-1}$. This constraint is known as the *precedence constraint* in the network scheduling literature [56]. In the following, we make a radical departure by relaxing this constraint to obtain a simpler single-hop virtual system, which will play a key role in designing our policy and its optimality analysis.

5.4.2 The Virtual Queue Process $\{\tilde{\boldsymbol{Q}}(t)\}_{t\geq 1}$

The *Virtual queue process* $\tilde{\boldsymbol{Q}}(t) = \big(\tilde{Q}_e(t), e \in E\big)$ is an $|E| = m$ dimensional controlled stochastic process, imitating a fictitious queueing network *without the precedence constraints*. In particular, when a packet p of class c arrives at the source node $s^{(c)}$, a dynamic policy π prescribes a suitable route $T^{(c)}(t) \in \mathcal{T}^{(c)}$ to the packet. Denoting the set of all edges in the route $T^{(c)}(t)$ by $\{l_1, l_2, \ldots, l_k\}$, this incoming packet induces a virtual arrival *simultaneously* at each of the virtual queues $\big(\tilde{Q}_{l_i}\big), i = 1, 2, \ldots, k$, right upon its arrival to the source. Since the virtual network is assumed to be relaxed with no precedence constraints, any packet present in the virtual queue is eligible for service. See Figure 5-1 for an illustration.

The (controlled) service process allocated to the virtual queues is denoted by

[2]Note that our notion of *virtual queues* is completely different from and unrelated to the notion of *shadow-queues* proposed earlier in [53], [23] and *virtual queues* proposed in [55].

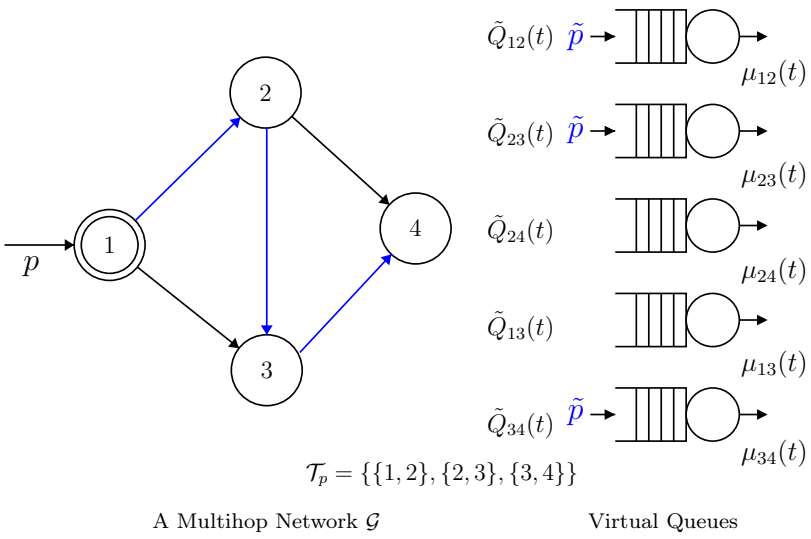


Figure 5-1: Illustration of the virtual queue system for the four-node network $\mathcal{G}$. Upon arrival, the incoming packet p, belonging to a unicast session from node 1 to 4, is prescribed a path $\mathcal{T}_p = \{\{1,2\},\{2,3\},\{3,4\}\}$. Relaxing the precedence constraints, the packet p is counted as an arrival to the virtual queues $\tilde{Q}_{12}$ and $\tilde{Q}_{23}$ and $\tilde{Q}_{34}$ simultaneously at the *same slot*. In the physical system, the packet p may take a while before reaching any edge in its path, depending on the control policy.

$\{\boldsymbol{\mu}^\pi(t)\}_{t\geq 1}$. We require the service process to satisfy the same activation constraints as in the original system, i.e., $\boldsymbol{\mu}^\pi(t) \in \mathcal{M}, \forall t \geq 1$.

Let $A_e^\pi(t)$ is the total number of virtual packet arrival (from all classes) to the queue $\tilde{Q}_e$ at time t under the action of the policy π, i.e.,

$$A_e^\pi(t) = \sum_{c\in\mathcal{C}} A^{(c)}(t)\mathbb{1}\left(e \in T^{(c)}(t)\right), \ \forall e \in E. \tag{5.3}$$

Hence, we have the following one-step evolution (Lindley recursion) of the virtual queue process $\{\tilde{Q}_e(t)\}_{t\geq 1}$:

$$\tilde{Q}_e(t+1) = \left(\tilde{Q}_e(t) + A_e^\pi(t) - \mu_e^\pi(t)\right)^+, \ \forall e \in E, \tag{5.4}$$

We emphasize that $A_e^\pi(t)$ is a function of the routing tree $T^{(c)}(t)$ that the policy chooses at time t, from the set of all admissible routes $\mathcal{T}^{(c)}$. This is discussed in the following.

5.4.3 Dynamic Control and Stability of the Virtual Queues

Next, we design a dynamic routing and link activation policy for the virtual network, which stabilizes the virtual queue process $\{\tilde{\boldsymbol{Q}}(t)\}_{t\geq 1}$, for all arrival rate-vectors $\boldsymbol{\lambda} \in \text{int}(\overline{\boldsymbol{\Lambda}})$. This policy is obtained by minimizing the one-step drift of a quadratic Lyapunov-function of the *virtual queue lengths* (as opposed to the real queue lengths used in the Back-Pressure policy [1]). In the following section, we will show that when this dynamic policy is used in conjunction with a suitable packet scheduling policy in the physical network, the overall policy is throughput-optimal for the physical network.

To derive a stabilizing policy for the virtual network, consider a quadratic Lyapunov function $L(\tilde{\boldsymbol{Q}}(t))$ defined in terms of the virtual queue lengths:

$$L(\tilde{\boldsymbol{Q}}(t)) = \sum_{e\in E} \tilde{Q}_e^2(t)$$

From the one-step dynamics of the virtual queues (5.4), we have:

$$\begin{aligned}\tilde{Q}_e(t+1)^2 &\leq (\tilde{Q}_e(t) - \mu_e^\pi(t) + A_e^\pi(t))^2 \\ &= \tilde{Q}_e^2(t) + (A_e^\pi(t))^2 + (\mu_e^\pi(t))^2 + 2\tilde{Q}_e(t)A_e^\pi(t) - 2\tilde{Q}_e(t)\mu_e^\pi(t) - 2\mu_e^\pi(t)A_e^\pi(t)\end{aligned}$$

Since $\mu_e^\pi(t) \geq 0$ and $A_e^\pi(t) \geq 0$, we have

$$\tilde{Q}_e^2(t+1) - \tilde{Q}_e^2(t) \leq (A_e^\pi(t))^2 + (\mu_e^\pi(t))^2 + 2\tilde{Q}_e(t)A_e^\pi(t) - 2\tilde{Q}_e(t)\mu_e^\pi(t)$$

Hence, the one-step Lyapunov drift $\Delta^\pi(t)$, conditional on the current virtual queue lengths $\tilde{\boldsymbol{Q}}(t)$, under the operation of any admissible Markovian policy $\pi \in \Pi$ is upper-bounded by

$$\begin{aligned}\Delta^\pi(t) &\stackrel{\text{def}}{=} \mathbb{E}\big(L(\tilde{\boldsymbol{Q}}(t+1)) - L(\tilde{\boldsymbol{Q}}(t))|\tilde{\boldsymbol{Q}}(t)\big) \\ &\leq B + 2\sum_{e\in E} \tilde{Q}_e(t)\mathbb{E}\big(A_e^\pi(t)|\tilde{\boldsymbol{Q}}(t)\big) - 2\sum_{e\in E} \tilde{Q}_e(t)\mathbb{E}\big(\mu_e^\pi(t)|\tilde{\boldsymbol{Q}}(t)\big) \qquad (5.5)\end{aligned}$$

where B is a constant, bounded by $\sum_e \mathbb{E}(A_e^\pi(t))^2 + \mathbb{E}(\mu_e^\pi(t))^2) \leq n^2 A_{\max}^2 + m$.
The upper-bound on the drift, given by (5.5), holds good for any admissible policy in the virtual network. In particular, by minimizing the upper-bound point wise, and exploiting the separable nature of the objective, we derive the following decoupled dynamic routing and link activation policy for the virtual network:

Dynamic Routing Policy

The optimal route for each class c, over the set of all admissible routes, is selected by minimizing the following cost function, appearing in the middle of Eqn. (5.5)

$$\mathsf{RoutingCost}^\pi \equiv \sum_{e \in E} \tilde{Q}_e(t) A_e^\pi(t),$$

where we remind the reader that $A_e^\pi(t)$ are the routing policy dependent arrivals to the virtual queue corresponding to the link e at time t.
Using Eqn. (5.3), we may rewrite the objective-function as

$$\mathsf{RoutingCost}^\pi = \sum_{c \in \mathcal{C}} A^{(c)}(t) \Bigg(\sum_{e \in E} \tilde{Q}_e(t) \mathbb{1}\big(e \in T^{(c)}(t)\big) \Bigg) \tag{5.6}$$

Using the separability of the objective (5.6), the above optimization problem decomposes into following min-cost route-selection problem $T_{\text{opt}}^{(c)}(t)$ for each class c:

$$T_{\text{opt}}^{(c)}(t) \in \arg \min_{T^{(c)} \in \mathcal{T}^{(c)}} \Bigg(\sum_{e \in E} \tilde{Q}_e(t) \mathbb{1}\big(e \in T^{(c)}\big) \Bigg) \tag{5.7}$$

Depending on the type of flow of class c, the optimal route-selection problem (5.7) is equivalent to one of the following well-known combinatorial problems on the graph $\mathcal{G}$, with its edges weighted by the virtual queue length vector $\tilde{\boldsymbol{Q}}$:

- **Unicast Traffic:** $T_{\text{opt}}^{(c)}(t)$ = The shortest $s^{(c)} - t^{(c)}$ path in the weighted-graph $\mathcal{G}$.

- **Broadcast Traffic:** $T_{\text{opt}}^{(c)}(t)$ = The minimum weight spanning tree rooted at the source $s^{(c)}$, in the weighted-graph $\mathcal{G}$.

- **Multicast Traffic:** $T_{\text{opt}}^{(c)}(t)$ = The minimum weight Steiner tree rooted at the source $s^{(c)}$ and spanning the destinations $\mathcal{D}^{(c)} = \{t_1^{(c)}, t_2^{(c)}, \ldots, t_k^{(c)}\}$, in the weighted-graph $\mathcal{G}$.

- **Anycast Traffic:** $T_{\text{opt}}^{(c)}(t)$ = The shortest of the k shortest $s^{(c)} - t_i^{(c)}$ paths, $i = 1, 2, \ldots, k$ in the weighted-graph $\mathcal{G}$.

Thus, the routes are selected according to a *dynamic source routing* policy [54]. Apart from the minimum weight Steiner tree problem for the multicast traffic (which is NP-hard with several known efficient approximation algorithms [57]), all of the above routing problems on the *weighted* virtual graph may be solved efficiently using standard algorithms [49].

Dynamic Link Activation Policy

A feasible link activation schedule $\boldsymbol{\mu}^*(t) \in \mathcal{M}$ is dynamically chosen at each slot by *maximizing* the last term in the upper-bound of the drift-expression (5.5), given as follows:

$$\boldsymbol{\mu}^*(t) \in \arg\max_{\boldsymbol{\mu} \in \mathcal{M}} \left(\sum_{e \in E} \tilde{Q}_e(t) \mu_e \right) \tag{5.8}$$

This is the well-known max-weight scheduling policy, which can be solved efficiently under various interference models (e.g., *Primary* or node-exclusive model [36]).
In solving the above routing and scheduling problems, we tacitly made the assumption that the virtual queue vector $\tilde{\boldsymbol{Q}}(t)$ is *globally known* at each slot. We will discuss practical distributed implementation of our algorithm in section 5.6.
Next, we establish stability of the virtual queues under the above policy, which will be instrumental for proving throughput-optimality of the overall **UMW** policy:

Theorem 5.4.1 *Under the above dynamic routing and link scheduling policy, the virtual queue process* $\{\tilde{\boldsymbol{Q}}(t)\}_{t\geq 0}$ *is strongly stable for any arrival rate* $\boldsymbol{\lambda} \in \text{int}(\overline{\boldsymbol{\Lambda}})$*, i.e.,*

$$\limsup_{T\to\infty} \frac{1}{T} \sum_{t=0}^{T-1} \sum_{e\in E} \mathbb{E}(\tilde{Q}_e(t)) < \infty.$$

Proof Consider an arrival rate vector $\boldsymbol{\lambda} \in \text{int}(\overline{\boldsymbol{\Lambda}})$. Thus, from Eqns. (5.1) and (5.2), it follows that there exists a scalar $\epsilon > 0$ and a vector $\boldsymbol{\mu} \in \text{conv}(\mathcal{M})$, such that we can decompose the total arrival for each class $c \in \mathcal{C}$ into a finite number of routes, such that

$$\lambda_e \overset{\text{(def.)}}{=} \sum_{(i,c):e\in T_i^{(c)}, T_i^{(c)}\in\mathcal{T}^{(c)}} \lambda_i^{(c)} \leq \mu_e - \epsilon, \quad \forall e \in E \tag{5.9}$$

By Caratheodory's theorem [39], we can write

$$\boldsymbol{\mu} = \sum_{i=1}^{m+1} p_i \boldsymbol{s}_i, \tag{5.10}$$

for some activation vectors $\boldsymbol{s}_i \in \mathcal{M}, \forall i$ and some probability distribution $\boldsymbol{p}$.

Now consider the following auxiliary stationary randomized routing and link activation policy **RAND** $\in \Pi$ for the virtual queue system $\{\tilde{\boldsymbol{Q}}(t)\}$, which will be useful in our proof. The randomized policy **RAND** randomly selects the activation vector $\boldsymbol{s}_j$ with probability $p_j, j = 1, 2, \ldots, m+1$ and routes the incoming packet of class c along the route $T_i^{(c)} \in \mathcal{T}^{(c)}$, with probability $\frac{\lambda_i^{(c)}}{\lambda^{(c)}}, \forall i, c$. Hence the total expected arrival rate to the virtual queue $\tilde{Q}_e$ at time slot t, due to the action of the stationary randomized policy **RAND** is given by

$$\mathbb{E}A_e^{\mathbf{RAND}}(t) = \lambda_e = \sum_{(i,c):e\in T_i^{(c)}, T_i^{(c)}\in\mathcal{T}^{(c)}} \lambda_i^{(c)}, \quad \forall e \in E \tag{5.11}$$

and the expected total service rate to the virtual server for the queue $\tilde{Q}_e$ is given by

$$\mathbb{E}\mu_e^{\mathbf{RAND}}(t) = \sum_{i=1}^{m+1} p_i \boldsymbol{s}_i(e) = \mu_e \tag{5.12}$$

Since our Max-Weight policy, **UMW**, maximizes the RHS of the drift expression in Eqn. (5.5) from the set of all feasible policies Π, we can write

$$\begin{aligned}
\Delta^{\mathbf{UMW}}(t) &\leq B + 2\sum_{e\in E} \tilde{Q}_e(t)\mathbb{E}\big(A_e^{\mathbf{RAND}}(t)|\tilde{\boldsymbol{Q}}(t)\big) - 2\sum_{e\in E} \tilde{Q}_e(t)\mathbb{E}\big(\mu_e^{\mathbf{RAND}}(t)|\tilde{\boldsymbol{Q}}(t)\big) \\
&\overset{(a)}{=} B + 2\sum_{e\in E} \tilde{Q}_e(t)\Big(\mathbb{E}A_e^{\mathbf{RAND}}(t) - \mathbb{E}\mu_e^{\mathbf{RAND}}(t)\Big) \\
&\overset{(b)}{=} B + 2\sum_{e\in E} \tilde{Q}_e(t)\big(\lambda_e - \mu_e\big) \\
&\overset{(c)}{\leq} B - 2\epsilon\sum_{e\in E} \tilde{Q}_e(t),
\end{aligned}$$

where (a) follows from the fact that the randomized policy **RAND** is memoryless and hence, independent of the virtual queues $\tilde{\boldsymbol{Q}}(t)$, (b) follows from Eqns. (5.11) and (5.12) and finally (c) follows from Eqn. (5.9).
Taking expectation of both sides w.r.t. the virtual queue lengths $\tilde{\boldsymbol{Q}}(t)$, we bound the expected drift at slot t as

$$\mathbb{E}L\big(\tilde{\boldsymbol{Q}}(t+1)\big) - \mathbb{E}L\big(\tilde{\boldsymbol{Q}}(t)\big) \leq B - 2\epsilon\sum_{e\in E}\mathbb{E}(\tilde{Q}_e(t)) \tag{5.13}$$

Summing Eqn. (5.13) from $t = 0$ to $T-1$ and remembering that $L(\boldsymbol{Q}(T)) \geq 0$ and $L(\tilde{\boldsymbol{Q}}(0)) = 0$, we conclude that

$$\frac{1}{T}\sum_{t=0}^{T-1}\sum_{e\in E}\mathbb{E}(\tilde{Q}_e(t)) \leq \frac{B}{2\epsilon} \tag{5.14}$$

Taking lim sup of both sides proves the claim.

As a consequence of the strong stability of the virtual queues $\{\tilde{Q}_e(t), e \in E\}$, we have the following sample-path result, which will be the key to our subsequent

analysis:

Lemma 5.4.2 *Under the action of the above policy, we have for any* $\boldsymbol{\lambda} \in \text{int}(\bar{\boldsymbol{\Lambda}})$*:*

$$\lim_{t\to\infty} \frac{\tilde{Q}_e(t)}{t} = 0, \ \forall e \in E, \quad \text{w.p. 1.}$$

In other words, the virtual queues are rate-stable [33].

Proof See Appendix 5.9.3.

The sample path result of Lemma 5.4.2 may be interpreted as follows: For any given realization ω of the underlying sample space Ω, define the function

$$F(\omega, t) = \max_{e\in E} \tilde{Q}_e(\omega, t).$$

Note that, for any $t \in \mathbb{Z}_+$, due to the boundedness of arrivals per slot, the function $F(\omega, t)$ is well-defined and finite. In view of this, Lemma (5.4.2) states that under the action of the **UMW** policy, $F(\omega, t) = o(t)$ *almost surely.* [3] This result will be used in our sample pathwise stability analysis of the physical queueing process $\{\boldsymbol{Q}(t)\}_{t\geq 0}$.

5.4.4 Consequence of the Stability of the Virtual Queues

It is apparent from the virtual queue evolution equation (5.4), that the stability of the virtual queues under the **UMW** policy implies that the arrival rate at each virtual queue is *at most* the service rate offered to it under the **UMW** routing and scheduling policy. In other words, *effective load* of each edge e in the virtual system is at most unity. This is a necessary condition for stability of the physical queues when the same routing and link activation policy is used for the multi-hop physical network. In the following, we make the notion of "effective-load" mathematically precise.

[3] $g(t) = o(t)$ if $\lim_{t\to\infty} \frac{g(t)}{t} = 0$.

Skorokhod Mapping Iterating on the system equation (5.4), we obtain the following well-known discrete time Skorokhod-Map representation [58] of the virtual queue dynamics

$$\tilde{Q}_e(t) = \left(\sup_{1 \leq \tau \leq t} \left(A_e^\pi(t-\tau, t) - S_e^\pi(t-\tau, t) \right) \right)^+, \tag{5.15}$$

where $A_e^\pi(t_1, t_2) \stackrel{\text{def}}{=} \sum_{\tau=t_1}^{t_2-1} A_e^\pi(\tau)$, is the total number of arrivals to the virtual queue $\tilde{Q}_e$ in the time interval $[t_1, t_2)$ and $S_e^\pi(t_1, t_2) \stackrel{\text{def}}{=} \sum_{\tau=t_1}^{t_2-1} \mu_e^\pi(\tau)$, is the total amount of service allocated to the virtual queue $\tilde{Q}_e$ in the interval $[t_1, t_2)$. For reference, we provide a proof of Eqn. (5.15) in Appendix 5.9.2.

Combining Equation (5.15) with Lemma 5.4.2, we conclude that under the **UMW** policy, *almost surely* for any sample path $\omega \in \Omega$, for each edge $e \in E$ and any $t_0 < t$, we have

$$A_e(\omega; t_0, t) \leq S_e(\omega; t_0, t) + F(\omega, t), \tag{5.16}$$

where $F(\omega, t) = o(t)$.

Implications for the Physical Network Note that, every packet arrival to a virtual queue $\tilde{Q}_e$ at time t corresponds to a packet in the physical network, that will eventually cross the edge e. Hence the loading condition (5.16) implies that under the **UMW** policy, the total number of packets injected during any time interval $(t_0, t]$, willing to cross the edge e, is less than the total amount of service allocated to the edge e in that time interval up to an additive term of $o(t)$. Thus informally, the "effective load" of any edge $e \in E$ is at most unity.

By utilizing the sample-path result in Eqn. (5.16), in the following section we show that there exists a simple packet scheduling scheme for the physical network, which guarantees the stability of the physical queues, and consequently, throughput-optimality.

5.5 Optimal Control of the Physical Network

With the help of the virtual queue structure as defined above, we next focus our attention on designing a throughput-optimal control policy for the multi-hop physical network. As discussed in Section 5.2, a control policy for the physical network consists of three components, namely (1) Routing, (2) Link activations and (3) Packet scheduling. In the proposed UMW policy, the (1) Routing and (2) Link activations for the physical network is done exactly in the same way as in the virtual network, based on the current values of the virtual queue state variables $\tilde{\boldsymbol{Q}}(t)$, described in Section 5.4.3. It is to be noted that, in the particular case of wireless networks, it is possible that a particular edge with positive virtual queue length is scheduled for transmission at a slot, even though the edge does not have any packet to transmit in its physical queue. The surprising fact, that follows from Theorem 6.5.3 is that this kind of wasted transmissions are rare and it *does not affect the throughput.*

There exist many possibilities for the third component, namely the packet scheduler, which efficiently resolves contention when multiple packets attempt to cross an active edge e at the same time-slot t. Popular choices for the packet scheduler include FIFO, LIFO etc. In this chapter, we focus on a particular scheduling policy which has its origin in the context of *adversarial queueing theory* [59]. In particular, we extend the *Nearest To Origin* (NTO) policy to the generalized network flow setting, where a packet may be duplicated. This policy was proposed in [60] in the context of wired networks for the unicast problem. We appropriately extend this policy for use in generalized flow problems, including multicast, broadcast, and anycast, even in wireless networks. Our proposed scheduling policy is called *Extended* NTO (**ENTO**) and is defined as follows:

Definition 5.5.1 (Extended NTO) *If multiple packets attempt to cross an active edge e at the same time slot t, the Extended Nearest To Origin (**ENTO**) policy gives priority to the packet which has traversed the least number of hops along its path from its origin up to the edge e.*

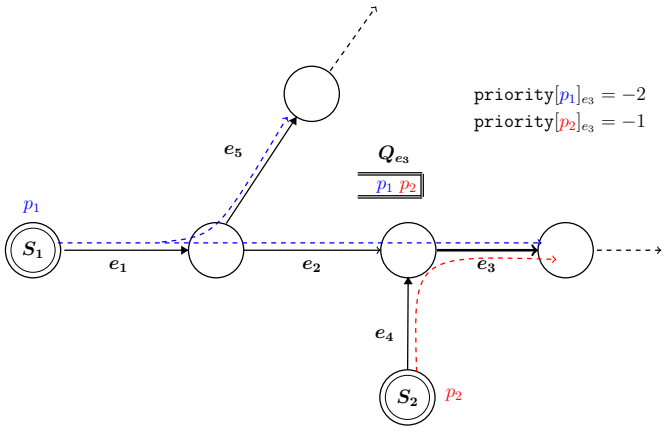


Figure 5-2: A schematic diagram showing the scheduling policy ENTO in action. The packets p_1 and p_2 originate from the sources S_1 and S_2. Part of their assigned routes are shown in blue and red respectively. The packets contend for crossing the active edge e_3 at the same time slot. According to the ENTO policy, the packet p_2 has higher priority (having crossed a single edge e_4 from its source) than p_1 (having crossed two edges e_1 and e_2 from its source) for crossing the edge e_3. Note that, although a copy of p_1 might have already crossed the edge e_5, this edge does not fall in the path connecting the source S_1 to the edge e_3 and hence does not enter into priority calculations.

The Extended NTO policy may be easily implemented by maintaining a single priority queue [49] per edge. The initial priority of each incoming packet at the source is set to zero. Upon transmission by any edge, the priority of a transmitted packet is decreased by one. The transmitted packet is then copied into the next-hop priority queue(s) (if any) according to its assigned route. See Figure 5-2 for an illustration. The pseudo code for the full UMW algorithm is provided in Algorithm 8.

We next state the following theorem which proves stability of the physical queues under the **ENTO** policy:

Theorem 5.5.2 *Under the action of the UMW policy with **ENTO** packet schedul-*

Algorithm 8 Universal Max-Weight Algorithm (UMW) at slot t for the Generalized Flow Problem in a Wireless Network

Require: Graph $\mathcal{G}(V, E)$, Virtual queue lengths $\{\tilde{Q}_e(t), e \in E\}$ at the slot t.

1: **[Edge-Weight Assignment]** Assign each edge of the graph $e \in E$ a weight $W_e(t)$ equal to $\tilde{Q}_e(t)$, i.e.

$$\boldsymbol{W}(t) \leftarrow \tilde{\boldsymbol{Q}}(t)$$

2: **[Route Assignment]** Compute a Minimum Weight Route $T^{(c)}(t) \in \mathcal{T}^{(c)}(t)$ for a class c incoming packet in the weighted graph $\mathcal{G}(V, E)$, according to Eqn. (5.7).

3: **[Link Activation]** Choose the activation $\boldsymbol{\mu}(t)$ from the set of all feasible activations $\mathcal{M}$, which maximizes the total activated link-weights, i.e.

$$\boldsymbol{\mu}(t) \leftarrow \arg\max_{\boldsymbol{s} \in \mathcal{M}} \boldsymbol{s} \cdot \boldsymbol{W}(t)$$

4: **[Packet Forwarding]** Forward physical packets from the physical queues over the activated links according to the **ENTO** scheduling policy.

5: **[Virtual Queue Counter Update]** Update the virtual queues assuming a precedence-relaxed system, *i.e.*,

$$\tilde{Q}_e(t+1) \leftarrow \left(\tilde{Q}_e(t) + A_e(t) - \mu_e(t)\right)^+, \ \forall e \in E$$

ing, the physical queues are rate-stable [33] for any arrival vector $\boldsymbol{\lambda} \in \text{int}(\overline{\boldsymbol{\Lambda}})$*, i.e.,*

$$\lim_{t\to\infty} \frac{\sum_{e\in E} Q_e(t)}{t} = 0, \quad \text{w.p.}\,1$$

Proof This theorem is proved by extending the argument of Gamarnik [60] and combining it with the sample path loading condition in Eqn. (5.16). See Appendix 6.8.3 for the detailed argument.

As a direct consequence of Theorem 6.5.2, we have the main result of this chapter:

Theorem 5.5.3 *The* UMW *policy, with ENTO packet scheduling rule, is throughput-optimal.*

Proof For any class $c \in \mathcal{C}$, the number of packets $R^{(c)}(t)$, received by all nodes $i \in \mathcal{D}^{(c)}$ may be bounded as follows:

$$A^{(c)}(0,t) - \sum_{e\in E} Q_e(t) \overset{(*)}{\leq} R^{(c)}(t) \leq A^{(c)}(0,t), \tag{5.17}$$

where the lower-bound $(*)$ follows from the simple observation that if a packet p of class c has not reached all destination nodes $\mathcal{D}^{(c)}$, then at least one copy of it must be present in some physical queue.
Dividing both sides of Eqn. (6.18) by t, taking limits and using SLLN and Theorem 6.5.2, we conclude that w.p. 1

$$\lim_{t\to\infty} \frac{R^{(c)}(t)}{t} = \lambda^{(c)}$$

Hence from the definition (1.1), we conclude that UMW is throughput-optimal.

5.6 Distributed Implementation

The **UMW** policy in its original form, as given in Algorithm 8, is centralized in nature. This is because the sources need to know the topology of the network and the current value of the virtual queues $\tilde{\boldsymbol{Q}}(t)$ to solve the shortest route and the Max-Weight problems at steps (2) and (3) of the algorithm. Although the topology of the network may be obtained efficiently by topology discovery algorithms [61], keeping track of the virtual queue evolution (Eqn. (6.4)) is subtle. Note that, in the special case where all packets arrive only at a single source node, no information exchange is necessary and the virtual queue updates (Step 5) may be implemented at the source locally. In the general case with multiple sources, it is necessary to periodically exchange packet arrival information among the sources to implement Step 5 exactly. To circumvent this issue, we propose the following class of heuristic **UMW** policies:

> [**Heuristic UMW**] Assign the edge weights to be the Physical queue lengths $\boldsymbol{Q}(t)$, instead of the virtual queue lengths $\tilde{\boldsymbol{Q}}(t)$, in *either* step (2) or step (3) or *both* in the original **UMW** Algorithm 8.

Routing based on physical queue lengths still requires the exchange of queue length information. However, this can be implemented efficiently using the standard distributed Bellman-Ford algorithm. The simulation results in section 5.7.2 show that the heuristic policy works well in practice and its delay performance is substantially better than the virtual queue based optimal **UMW** policy in wireless networks. The affirmative simulation results shown in the next section immediately prompts us to make the following conjecture:

Conjecture 2 *The Heuristic UMW policy is throughput-optimal.*

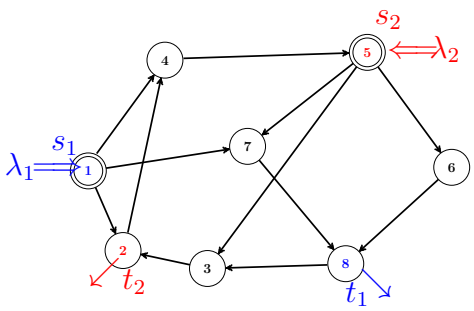


Figure 5-3: The wired network topology used for unicast simulation

5.7 Numerical Simulation

5.7.1 Delay Improvement Compared to the Back Pressure Policy - the Unicast Setting

To empirically demonstrate the superior performance of the UMW policy over the Back-Pressure class of policies in the unicast setting, we consider the wired network shown in Figure 5-3 and implement the following policies: (1) UMW (opt), (2) UMW (heuristic), (3) Backpressure (original [1]), and (4) Shortest-Path based Backpressure [25].

All links are assumed to have a unit capacity. We consider two concurrent unicast sessions with source-destination pairs given by $(s_1 = 1, t_1 = 8)$ and $(s_2 = 5, t_2 = 2)$ respectively. It is easy to see that Max-Flow$(s_1 \to t_1) = 2$ and Max-Flow$(s_2 \to t_2) = 1$ and there exist mutually disjoint paths to achieve the optimal rate-pair $(\lambda_1, \lambda_2) = (2, 1)$. Assuming Poisson arrivals at the sources s_1 and s_2 with intensities $\lambda_1 = 2\rho$ and $\lambda_2 = \rho$, $0 \leq \rho \leq 1$, where ρ denotes the "load factor", Figure 5-4 shows a plot of total average queue lengths as a function of the load factor ρ under the operation of the four policies considered above.

From the plot, we conclude that both the optimal and heuristic UMW policies *unformly* outperform the **BP** (original) and SP-based BP policy in terms of average queue lengths, and hence (by Little's Law), end-to-end delay. The primary reason being, the **BP** class of policies, in principle, explores all possible paths to route packets to their destinations. The UMW policy, on the other hand, transmits all packets along "optimal" acyclic routes. This results in substantial reduction in latency.

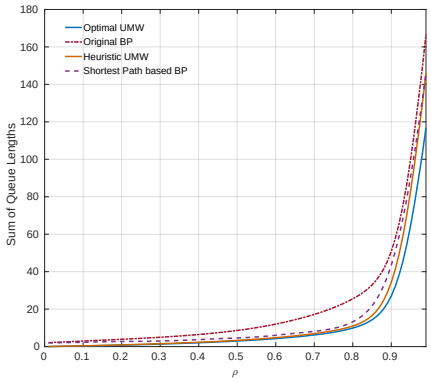


Figure 5-4: Comparison of time-averaged queue-lengths under the **BP** (original and shortest-path based) and UMW (optimal and heuristic) policies in the unicast setting of Fig. 5-3. In terms of performance, we have UMW (opt)> UMW (heu) > BP (SP-based) > BP (original).

5.7.2 Using the Heuristic UMW policy for Improved Latency in the Wireless Networks - the Broadcast Setting

Next, we empirically demonstrate that the heuristic UMW policy that uses physical queue lengths $\boldsymbol{Q}(t)$ (instead of virtual queues $\tilde{\boldsymbol{Q}}(t)$ as in the optimal UMW policy) not only achieves the full broadcast capacity but yields better delay performance in this particular wireless network. As discussed earlier, the heuristic policy is practically easier to implement in a distributed fashion. We simulate a 3×3 wireless grid network shown in Figure 5-5, with *primary* interference constraints [3]. The broadcast capacity of the network is known to be $\lambda^* = \frac{2}{5}$ [62]. The ENTO policy is used for packet scheduling. The average queue length is plotted in Figure 5-6 as a function of the packet arrival rate λ under the operation of the (a) UMW (optimal) and (b) UMW (heuristic) policies. The plot shows that the heuristic policy results in much smaller queue lengths than the optimal policy. The reason being that physical queues capture the network congestion “more accurately" for proper link activations.

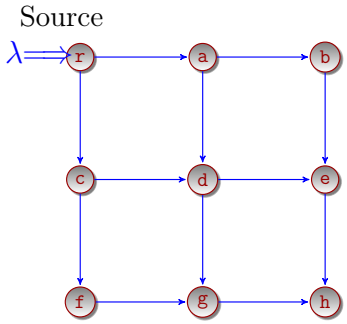


Figure 5-5: The wireless topology used for broadcast simulation

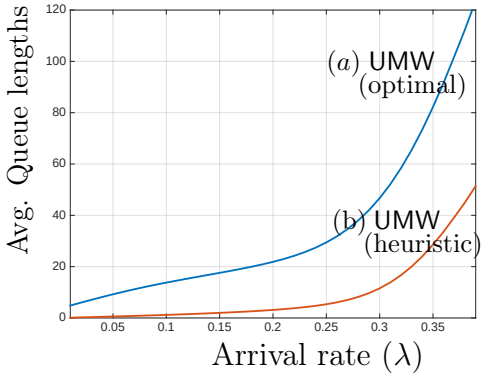


Figure 5-6: Comparison of the Avg. Queue lengths as a function of the arrival rate for the optimal (in blue) and the heuristic (in red) UMW Policy for the grid network in Figure 5-5 in the **broadcast** setting.

5.7.3 Performance of the Optimal and Heuristic UMW Policy in Time-varying networks- the Broadcast Setting

In this section, we take a closer look at the wireless grid network in Figure 5-5 by numerically evaluating the broadcasting performance of the proposed policies, when the network is time-varying. In particular, we assume that at each slot a link is ON with probability p_{ON}, and is OFF w.p. $1 - p_{\text{ON}}$, independent of everything else. Packets can be transmitted only over the ON links at a given slot. Using similar analysis that we did for the static network, it can be easily shown that the proposed UMW policy remains throughput-optimal when the Max-Weight link activation at each slot is done with respect to the ON links at that slot. The packet routing policy remains the same as in the original UMW policy. The performance of the optimal and heuristic UMW policy is shown in Figure 5-7 for two different values of the parameter p_{ON}. It can be seen from the plot that the heuristic policy incurs substantially smaller queue lengths, compared to the optimal policy, especially in the low-load regime. Also, from the nearly identical vertical asymptotes in the queue length vs arrival rate plots, we conclude that the heuristic policy is also throughput-optimal in this case.

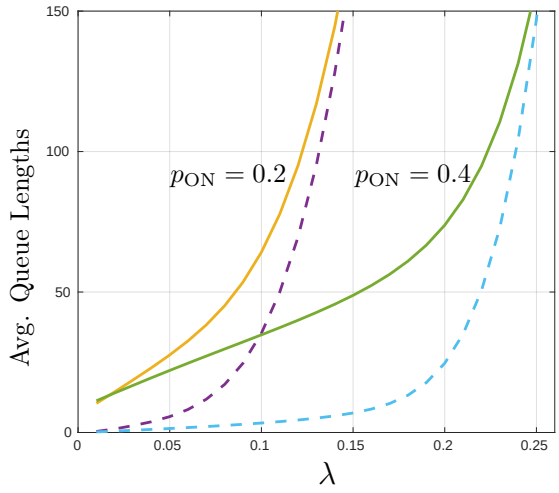


Figure 5-7: Comparison of the time-averaged total queue lengths under the optimal (solid line) and heuristic (dashed line) UMW policy in the time-varying grid network (with parameter p_{ON}), for the broadcast problem.

5.8 Conclusion

In this chapter, we have proposed a new, efficient and throughput-optimal policy, named **U**niversal **M**ax-**W**eight (**UMW**), for the Generalized Network Flow problem. The **UMW** policy can simultaneously handle a mix of Unicast, Broadcast, Multicast and Anycast traffic in arbitrary networks and is empirically shown to have superior performance compared to the existing policies. The next step would be to investigate whether the **UMW** policy still retains its optimality when implemented with physical queue lengths, instead of the virtual queue lengths. An affirmative answer to this question would imply a more efficient implementation of the policy.

5.9 Appendix

5.9.1 Proof of Converse of Theorem 5.2.1

Proof Consider any admissible arrival rate vector $\boldsymbol{\lambda} \in \boldsymbol{\Lambda}(\mathcal{G}, \mathcal{C})$. By definition, there exists an admissible policy $\pi \in \Pi$ which supports the arrival vector $\boldsymbol{\lambda}$ in the sense of Eqn. (1.1). Without any loss of generality, we may assume the policy π to be stationary and the associated DTMC to be ergodic. Let $A_i^{(c)}(t)$ denote the total

number of packets from class c that have finished their routing along the route $T_i^{(c)} \in \mathcal{T}^{(c)}$ up to time t. Note that, each packet is routed along one admissible route only. Hence, if the total number of arrival to the source $s^{(c)}$ of class c up to time t is denoted by the random variable $A^{(c)}(t)$, we have

$$A^{(c)}(t) \overset{(a)}{\geq} \sum_{T_i^{(c)} \in \mathcal{T}^{(c)}} A_i^{(c)}(t) \overset{(b)}{=} R^{(c)}(t). \tag{5.18}$$

In the above, the inequality (a) follows from the observation that any packet p which has finished its routing along some route $T_i^{(c)} \in \mathcal{T}^{(c)}$ by the time t, must have arrived at the source by the time t. The equality (b) follows from the observation that any packet p which has finished its routing by time t along some route $T_i^{(c)} \in \mathcal{T}^{(c)}$, has reached all of the destination nodes $\mathcal{D}^{(c)}$ of class c by time t and vice versa.
Dividing both sides of equation (5.18) by t and taking limit as $t \to \infty$, we have $w.p.1$

$$\begin{aligned} \lambda^{(c)} \overset{(d)}{=} \lim_{t\to\infty} \frac{A^{(c)}(t)}{t} &\geq \liminf_{t\to\infty} \frac{1}{t} \sum_{T_i^{(c)} \in \mathcal{T}^{(c)}} A_i^{(c)}(t) \\ &= \liminf_{t\to\infty} \frac{R^{(c)}(t)}{t} \\ &\overset{(f)}{=} \lambda^{(c)}, \end{aligned}$$

where equality (d) follows from the SLLN, and equality (f) follows from the Definition (1.1).
From the above inequalities, we conclude that w.p. 1

$$\lim_{t\to\infty} \frac{1}{t} \sum_{T_i^{(c)} \in \mathcal{T}^{(c)}} A_i^{(c)}(t) = \lambda^{(c)}, \quad \forall c \in \mathcal{C} \tag{5.19}$$

Now we use the fact that the policy π is stationary and the associated DTMC is *ergodic*. Thus the time-average limits exist and they are constant *a.s.*. For all $T_i^{(c)} \in \mathcal{T}^c$ and $c \in \mathcal{C}$, define

$$\lambda_i^{(c)} \overset{\text{def}}{=} \lim_{t\to\infty} \frac{1}{t} A_i^{(c)}(t) \tag{5.20}$$

Hence, from the above, we get

$$\lambda^{(c)} = \sum_{T_i^{(c)} \in \mathcal{T}^{(c)}} \lambda_i^{(c)} \tag{5.21}$$

Now consider any edge $e \in E$ in the graph $\mathcal{G}$. Since the variable $A_i^{(c)}(t)$ denotes the total number of packets from class c, that have *completely* traversed along the tree $T_i^{(c)}$, the following inequality holds good for any time t

$$\sum_{(i,c):e \in T_i^{(c)}, T_i^{(c)} \in \mathcal{T}^{(c)}} A_i^{(c)}(t) \leq \sum_{\tau=1}^{t} \mu_e(\tau), \tag{5.22}$$

where the left-hand side denotes a lower-bound on the number of packets that have crossed the edge e and the right hand side denotes the amount of service that have been provided to edge e up to time t by the policy π.
Dividing both sides by t and taking limits of both side, and noting that the limit on the left-hand side exists w.p. 1, we have

$$\sum_{(i,c):e \in T_i^{(c)}, T_i^{(c)} \in \mathcal{T}^{c}} \lambda_i^{(c)} \leq \overline{\mu}_e, \tag{5.23}$$

where $\overline{\boldsymbol{\mu}} = \lim_{t\to\infty} \frac{1}{t} \sum_{\tau=1}^{t} \boldsymbol{\mu}(\tau)$. Since $\boldsymbol{\mu}(\tau) \in \mathcal{M}, \forall \tau$ and the set $\text{conv}(\mathcal{M})$ is closed, we conclude that $\overline{\boldsymbol{\mu}} \in \text{conv}(\mathcal{M})$. Eqns. (5.21) and (5.23) concludes the proof of the theorem.

5.9.2 Proof of the Skorokhod Map Representation in Eqn. (5.15)

Proof From the dynamics of the virtual queues in Eqn. (6.4), we have for any $t \geq 1$

$$\tilde{Q}_e(t) \geq \tilde{Q}_e(t-1) + A_e(t-1) - \mu_e(t-1). \tag{5.24}$$

Iterating (5.24) τ times $1 \leq \tau \leq t$, we obtain

$$\tilde{Q}_e(t) \geq \tilde{Q}_e(t-\tau) + A_e(t-\tau, t) - S_e(t-\tau, t),$$

where $A_e(t_1, t_2) = \sum_{\tau=t_1}^{t_2-1} A_e(\tau)$ and $S_e(t_1, t_2) = \sum_{\tau=t_1}^{t_2-1} \mu_e(\tau)$, as defined before. Since each of the virtual-queue components are non-negative at all times (viz. (6.4)), we have $\tilde{Q}_e(t-\tau) \geq 0$. Thus,

$$\tilde{Q}_e(t) \geq A_e(t-\tau, t) - S_e(t-\tau, t).$$

Since the above holds for any time $1 \leq \tau \leq t$ and the queues are always non-negative, we obtain

$$\tilde{Q}_e(t) \geq \left(\sup_{1 \leq \tau \leq t} \left(A_e(t-\tau, t) - S_e(t-\tau, t) \right) \right)_+ \tag{5.25}$$

To show that Eqn. (5.25) holds with equality, we consider two cases.

Case I: $\tilde{Q}_e(t) = 0$

Since the RHS of Eqn. (5.25) is non-negative, we immediately obtain equality throughout in Eqn (5.25).

Case II: $\tilde{Q}_e(t) > 0$ Consider the *latest time* $t - \tau', 1 \leq \tau' \leq t$, prior to t, at which $\tilde{Q}_e(t-\tau') = 0$. Such a time $t - \tau'$ exists because we assumed the system to start with empty queues at time $t = 0$. Hence $Q_e(z) > 0$ throughout the time interval $z \in [t - \tau' + 1, t]$. As a result, in this time interval the system dynamics for the virtual-queues (6.4) takes the following form

$$\tilde{Q}_e(z) = \tilde{Q}_e(z-1) + A_e(z-1) - \mu_e(z-1),$$

Iterating the above recursion in the interval $z \in [t - \tau' + 1, t]$, we obtain

$$\tilde{Q}_e(t) = A_e(t - \tau', t) - S_e(t - \tau', t) \tag{5.26}$$

We conclude the proof upon combining Eqns. (5.25) and (5.26).

5.9.3 Proof of Lemma 5.4.2

Proof We will establish this result by appealing to the *Strong Stability Theorem* (Theorem 2.8) of [33]. For this, we first consider an associated system $\{\hat{\boldsymbol{Q}}(t)\}_{t\geq 0}$ with a slightly different queueing recursion, as considered in [33] (Eqn. 2.1, pp-15). For a given sequence $\{\boldsymbol{A}(t), \boldsymbol{\mu}(t)\}_{t\geq 0}$, define the following recursion for all $e \in E$,

$$\begin{aligned} \hat{Q}_e(t+1) &= (\hat{Q}_e(t) - \mu_e(t))_+ + A_e(t), \\ \hat{Q}_e(0) &= 0. \end{aligned} \tag{5.27}$$

Recall the dynamics of the virtual queues (Eqn. (6.4)):

$$\begin{aligned} \tilde{Q}_e(t+1) &= (\tilde{Q}_e(t) + A_e(t) - \mu_e(t))^+, \\ \tilde{Q}_e(0) &= 0. \end{aligned} \tag{5.28}$$

We next prove the following proposition:

Proposition 5.9.1 *For all $e \in E$*

$$A_{\max} + \tilde{Q}_e(t) \overset{(*)}{\geq} \hat{Q}_e(t) \overset{(**)}{\geq} \tilde{Q}_e(t), \quad \forall t \geq 0.$$

Proof We first prove the second inequality (**) by inducting on time.

Base Step $t = 0$:

Holds with equality since $\hat{Q}_e(0) = \tilde{Q}_e(0) = 0$.

Induction Step:

Assume that $\hat{Q}_e(t) \geq \tilde{Q}_e(t)$ for some $t \geq 0$. From the dynamics (5.27), we can write

$$\begin{aligned}
\hat{Q}_e(t+1) &= \max\left(\hat{Q}_e(t) - \mu_e(t) + A_e(t), A_e(t)\right) \\
&\overset{(a)}{\geq} \max\left(\hat{Q}_e(t) - \mu_e(t) + A_e(t), 0\right) \\
&\overset{(b)}{\geq} \max\left(\tilde{Q}_e(t) - \mu_e(t) + A_e(t), 0\right) \\
&\overset{(c)}{=} \tilde{Q}_e(t+1),
\end{aligned}$$

where Eqn. (a) follows from the fact that $A_e(t) \geq 0$, Eqn. (b) follows from the induction assumption and Eqn. (c) follows from the dynamics (5.28). This completes the induction step and the proof of the second inequality (**) of the proposition. Proof of the first inequality (*) may also be carried out similarly.

Taking expectation throughout the first inequality (*) of Proposition 5.9.1 for any $e \in E$, we have for each $t \geq 0$

$$\mathbb{E}(\hat{Q}_e(t)) \leq \mathbb{E}(\tilde{Q}_e(t)) + A_{\max}$$

Thus,

$$\begin{aligned}
\limsup_{T\to\infty} \frac{1}{T}\sum_{t=0}^{T-1} \mathbb{E}(\hat{Q}_e(t)) &\leq \limsup_{T\to\infty} \frac{1}{T}\sum_{t=0}^{T-1} \mathbb{E}(\tilde{Q}_e(t)) + A_{\max} \\
&\overset{(a)}{<} \infty,
\end{aligned}$$

where (a) follows from the strong stability of the virtual queues under **UMW**. This shows that, the associated queue process $\{\hat{\boldsymbol{Q}}(t)\}_{t\geq 0}$ is also strongly stable under **UMW**.

Since the total external arrival $A(t) = \sum_e A_e(t)$ at slot t is assumed to be bounded w.p. 1, applying Theorem 2.8, part (b) of [33], we conclude that for any $e \in E$

$$\lim_{t\to\infty} \frac{\hat{Q}_e(t)}{t} = 0, \quad \text{w.p.1}$$

Using the second inequality (**) of Proposition 5.9.1 and the non-negativity of the virtual queues, we conclude that for any $e \in E$

$$\lim_{t\to\infty} \frac{\tilde{Q}_e(t)}{t} = 0, \quad \text{w.p.1}$$

Finally, using the union bound we conclude that

$$\lim_{t\to\infty} \frac{\tilde{Q}_e(t)}{t} = 0, \ \forall e \in E \quad \text{w.p.1}$$

5.9.4 Proof of Theorem 6.5.2

Throughout this proof, we will fix a sample point $\omega \in \Omega$, giving rise to a sample path satisfying the condition (5.16). All random processes [4] will be evaluated at this sample path. For the sake of notational simplicity, we will drop the argument ω for evaluating any random variable $\boldsymbol{X}$ at the sample point ω, e.g., the deterministic sample-path $X(\omega, t)$ will be simply denoted by $X(t)$. We now establish a simple analytical result which will be useful in the main proof of the theorem:

Lemma 5.9.2 *Consider a non-negative function $\{F(t), t \geq 1\}$ defined on the set of natural numbers, such that $F(t) = o(t)$. Define $M(t) = \sup_{0\leq\tau\leq t} F(\tau)$. Then*

1. *$M(t)$ is non-decreasing in t.*
2. *$M(t) = o(t)$*

Proof That $M(t)$ is non-decreasing follows directly from the definition of $M(t) = \sup_{0\leq\tau\leq t} F(t)$. We now prove the claim (2).

Case I: The function $F(t)$ is bounded

In this case, the function $M(t)$ is also bounded and the claim follows immediately.

Case II: The function $F(t)$ is unbounded

[4] Recall that, a discrete-time integer-valued random process $\boldsymbol{X}(\omega; t)$ is a measurable map from the sample space Ω to the set of all integer-sequences $\mathbb{Z}^\infty$ [50], i.e., $\boldsymbol{X} : \Omega \to \mathbb{Z}^\infty$.

Define the subsequence $\{r_k\}_{k\geq 1}$, corresponding to the time of maximums of the function $M(t)$ up to time t. Formally the sequence $\{r_k\}_{k\geq 1}$ is defined recursively as follows,

$$r_1 = 1 \tag{5.29}$$

$$r_k = \{\min t > r_{k-1} : F(t) > \max_{\tau \leq t-1} F(\tau)\} \tag{5.30}$$

Since the function $F(t)$ is assumed to be unbounded, we have $r_k \to \infty$ as $k \to \infty$. In the literature [63], the sequence $\{r_k\}$ is also known as the sequence of *records* of the function $F(t)$. With this definition, for any $t \geq 1$ and for $r_k \leq t$ corresponding to the latest record up to time t, we readily have

$$M(t) = F(r_k) \tag{5.31}$$

Hence,

$$\frac{M(t)}{t} = \frac{F(r_k)}{t} \overset{(a)}{\leq} \frac{F(r_k)}{r_k}, \tag{5.32}$$

where Eqn. (a) follows from the fact that $r_k \leq t$. Thus for any sequence of natural numbers $\{t_i\}_1^\infty$, we have a corresponding sequence $\{r_{k_i}\}_{i=1}^\infty$ such that for each i, we have

$$\frac{M(t_i)}{t_i} = \frac{F(r_{k_i})}{t} \overset{(a)}{\leq} \frac{F(r_{k_i})}{r_{k_i}}$$

This implies,

$$\limsup_{t\to\infty} \frac{M(t)}{t} \leq \limsup_{t\to\infty} \frac{F(t)}{t} \overset{(b)}{=} 0, \tag{5.33}$$

where Eqn (b) follows from our hypothesis on the function $F(t)$. Also since $M(t) \geq F(t)$, from Eqn. (5.33) we conclude that

$$\lim_{t\to\infty} \frac{M(t)}{t} = 0 \tag{5.34}$$

As a direct consequence of Lemma 6.8.6 and the property of the sample-point ω under consideration, we have:

$$A_e(t_0, t) \leq S_e(t_0, t) + M(t), \quad \forall e \in E, \, \forall t_0 \leq t \tag{5.35}$$

for some non-decreasing non-negative function $M(t) = o(t)$. Equipped with Eqn. (6.24), we return to the proof of the Theorem 6.5.2.

Proposition 5.9.3 *ENTO is rate-stable.*

Proof We generalize the argument by Gamarnik [60] to prove the proposition. We remind the reader that we are analyzing the time-evolution of a fixed sample point $\omega \in \Omega$, which satisfies Eqn. (6.24).

Let $R_e(0)$ denote the total number of packets waiting to cross the edge e at time $t = 0$. Also, let $R_k(t)$ denote the total number of packets at time t, which are *exactly* k hops away from their respective sources. Such packets will be called "layer k" packets in the sequel. If a packet is duplicated along its assigned route T (which is, in general, a tree), each copy of the packet is counted separately in the variable $R_k(t)$, *i.e.,*

$$R_k(t) = \sum_{T \in \mathcal{T}} R_{(e_k^T, T)}(t), \tag{5.36}$$

where the variable $R_{(e,T)}(t)$ denotes the number of packets following the routing tree T, that are waiting to cross the edge $e \in T$ at time t. The edge e_k^T is an edge located k^{th} hop away from the source in the tree T. If there are more than one such edge (because the tree T has more than one branch), we include all these edges in the summation (6.25). We show by induction that $R_k(t)$ is *almost surely* bounded by a

function, which is $o(t)$.

Base Step $k = 0$: Fix an edge e and time t. Let $t_0 \leq t$ be the largest time at which no packets of layer 0 (packets which have not crossed any edge yet) were waiting to cross e. If no such time exists, set $t_0 = 0$. Hence, the total number of layer 0 packets waiting to cross the edge e at time t_0 is at most $Q_e(0)$. During the time interval $[t_0, t]$, as a consequence of the **UMW** control policy (6.24), at most $S_e(t_0, t) + M(t)$ external packets have been admitted to the network, that want to cross the edge e in future. Also, by the choice of the time t_0, the edge e was always having packets to transmit during the entire time interval $[t_0, t]$. Since ENTO scheduling policy is followed, layer 0 packets have priority over all other packets. Hence, it follows that the total number of packets at the edge e at time t satisfies

$$\begin{aligned} \sum_{T: e \in e_0^T} R_{(e,T)}(t) &\leq R_e(0) + S_e(t_0, t) + M(t) - S_e(t_0, t) \\ &\leq R_e(0) + M(t) \end{aligned} \tag{5.37}$$

As a result, we have $R_0(t) \leq \sum_e R_e(0) + |E| M(t)$, for all t. Let $B_0(t) \stackrel{\text{def}}{=} \sum_e R_e(0) + |E| M(t)$. Since $M(t) = o(t)$, we have $B_0(t) = o(t)$. Note that, since $M(t)$ is monotonically non-decreasing by definition, so is $B_0(t)$.

Induction Step: Suppose that, for some monotonically non-decreasing functions $B_j(t) = o(t), j = 0, 1, 2, \ldots, k-1$, we have $R_j(t) \leq B_j(t)$, for all time t. We next show that $R_k(t) \leq B_k(t)$ for all t, where $B_k(t) = o(t)$.

Again, fix an edge e and an arbitrary time t. Let $t_0 \leq t$ denote the largest time before t, such that there were no layer k packets waiting to cross the edge e. Set $t_0 = 0$ if no such time exists. Hence the edge e was always having packets to transmit during the time interval $[t_0, t]$ (packets in layer k or lower). The layer k packets that wait to cross edge e at time t are composed only of a subset of packets which were in layers $0 \leq j \leq k-1$ at time t_0 or packets that arrived during the time interval $[t_0, t]$ and have edge e as *one of their* k^{th} *edge* on the route followed. By our induction assumption, the first group of packets has a size bounded by $\sum_{j=0}^{k-1} B_j(t_0) \leq \sum_{j=0}^{k-1} B_j(t)$, where we have used the fact (from our previous induction step) that the functions $B_j(\cdot)$'s

are monotonically non-decreasing. The size of the second group of packets is given by $\sum_{T:e\in e_k^T} A_T(t_0,t)$. We next estimate the number of layer k packets that crossed the edge e during the time interval $[t_0,t]$. Since ENTO policy is used, layer k packets were not processed only when there were packets in layers up to $k-1$ that wanted to cross e. The number of such packets is bounded by $\sum_{j=0}^{k-1} B_j(t_0) \leq \sum_{j=0}^{k-1} B_j(t)$, which denotes the total possible number of packets in layers up to $k-1$ at time t_0, plus $\sum_{j=0}^{k-1}\sum_{T:e\in e_j^T} A_T(t_0,t)$, which is the number of new packets that arrived in the interval $[t_0,t]$ and intend to cross the edge e within first $k-1$ hops. Thus, we conclude that at least

$$\max\left\{0, S_e(t_0,t) - \sum_{j=0}^{k-1} B_j(t) - \sum_{j=0}^{k-1}\sum_{T:e\in e_j^T} A_T(t_0,t)\right\} \tag{5.38}$$

packets of layer k crossed e during the time interval $[t_0,t]$. Hence,

$$\begin{aligned}
\textstyle\sum_{T:e\in e_k^T} R_{(e,T)}(t) &\leq \sum_{j=0}^{k-1} B_j(t) + \sum_{T:e\in e_k^T} A_T(t_0,t) \\
- &\quad \big(S_e(t_0,t) - \sum_{j=0}^{k-1} B_j(t) - \sum_{j=0}^{k-1}\sum_{T:e\in e_j^T} A_T(t_0,t)\big) \\
= &\quad 2\sum_{j=0}^{k-1} B_j(t) + \sum_{j=0}^{k}\sum_{T:e\in e_j^T} A_T(t_0,t) - S_e(t_0,t) \\
\overset{(a)}{\leq} &\quad 2\sum_{j=0}^{k-1} B_j(t) + M(t),
\end{aligned}$$

where Eqn. (a) follows from the arrival condition (6.24). Hence the total number of layer k packets at time t is bounded by

$$R_k(t) \leq 2|E|\sum_{j=0}^{k-1} B_j(t) + M(t)|E| \tag{5.39}$$

Define $B_k(t)$ to be the RHS of the above equation, i.e.

$$B_k(t) \stackrel{(\text{def})}{=} 2|E| \sum_{j=0}^{k-1} B_j(t) + M(t)|E| \tag{5.40}$$

Using our induction assumption and Eqn. (6.28), we conclude that $B_k(t) = o(t)$ and it is monotonically non-decreasing. This completes the induction step.

To conclude the proof of the proposition, notice that total size of the physical queues at time t may be alternatively written as

$$\sum_{e \in E} Q_e(t) = \sum_{k=1}^{n-1} R_k(t) \tag{5.41}$$

Since the previous inductive argument shows that for all k, we have $R_k(t) \leq B_k(t)$ where $B_k(t) = o(t)$ *a.s.*, we conclude that

$$\lim_{t \to \infty} \frac{\sum_{e \in E} Q_e(t)}{t} = 0, \quad \text{w.p. } 1, \tag{5.42}$$

This implies that the physical queues are rate stable [33], jointly under the operation of **UMW** and ENTO.

Chapter 6

Throughput-Optimal Broadcast in Wireless Networks with Point-to-Multipoint Transmissions

6.1 Overview of the Results

A fundamental feature of the wireless medium is the inherent **point-to-mutipoint** nature of wireless links, where a packet transmitted by a node can be heard by all its neighbors. This feature, also known as the wireless broadcast advantage [20], is especially useful in network-wide broadcast applications, where the objective is to efficiently disseminate the packets among all nodes in the network. Additionally, because of inter-node interference, the set of simultaneous transmissions in a wireless network is restricted to the set of non-interfering feasible schedules. In the previous chapters, we designed a number of throughput-optimal broadcasting policies for wireless networks with **point-to-point** links. Designing a broadcast algorithm which efficiently utilizes the broadcast advantage, while respecting the interference constraints is a more challenging problem. In this chapter we solve this problem by leveraging the tools and algorithmic techniques developed in Chapter 5.

The main contributions of this chapter are as follows:

- We propose an online dynamic policy for throughput-optimal broadcasting in wireless networks with point-to-multipoint links.

- We prove the **NP**-completeness of the corresponding finite horizon wireless broadcast problem.

- We introduce a new control policy and proof technique by combining the stochastic Lyapunov drift theory with the deterministic adversarial queueing theory. This essentially enables us to derive a stabilizing control policy for a multi-hop network by solving the problem on a simpler *precedence-relaxed* virtual single-hop network.

The rest of the chapter is organized as follows. In section 6.2 we describe the system model and formulate the problem. In section 6.3 we prove the hardness of the finite-horizon version of the problem. Next, in section 6.4 we derive an optimal control policy for a related relaxed version of the wireless network. This control policy is then applied to the original unrelaxed network in section 6.5, where we show that the resulting policy is throughput-optimal, when used in conjunction with a priority-based packet scheduling policy. In section 6.6, we demonstrate the efficacy of the proposed policy via numerical simulations. Finally, we conclude the chapter in section 6.7.

6.2 System Model and Problem Formulation

We consider the problem of efficiently disseminating packets, arriving randomly at source nodes, to all nodes in a wireless network. The system model and the precise problem statement are described below.

6.2.1 Network Model

Consider a wireless network with its topology given by the directed graph $\mathcal{G}(V, E)$. The set V denotes the set of all nodes, with $|V| = n$. If node j is within the

transmission range of node i, there is a directed edge $(i, j) \in E$ connecting them. Due to the inherent point-to-multi-point broadcast nature of the radio channel, a transmitted packet can be heard by all out-neighbors of the transmitting node. In other words, the packets are transmitted over the *hyperedges*, where a hyperedge is defined to be the union of all outgoing edges from a node. The system evolves in a slotted time structure. External packets, which are to be broadcasted throughout the network, arrive at designated source nodes. Total number of external packet arrivals at any slot is assumed to be bounded by a finite constant.

For simplicity of exposition, we consider only static networks with a single source node `r`. However, the algorithm and its analysis presented in this chapter extend to time-varying dynamic networks with multiple source nodes in a straightforward manner. We will consider time-varying networks in our numerical simulations.

6.2.2 Wireless Transmission Model

When a node $i \in V$ is scheduled for transmission, it can transmit any of its received packets at the rate of c_i packets per slot to *all* of its out-neighbors over its outgoing hyperedge. See Figure 6-1. Due to the wireless interference constraint, only a selected subset of nodes can feasibly transmit over the hyperedges simultaneously without causing collisions. The wireless channel is assumed to be error-free otherwise. The set of all feasible transmission schedules may be described concisely using the notion of a *Conflict Graph* $\mathcal{C}(\mathcal{G})$. The set of vertices in the conflict graph is the same as the set of nodes in the network V. There is an edge between two nodes in the conflict graph if and only if these two nodes *cannot* transmit simultaneously without causing collision. Note that our *node-centric* definition of conflict graphs is a little different from the traditional *edge-centric* definition of conflict graph, which concerns point-to-point transmissions [64] [65].

As the simplest example of the interference model, consider a wireless network where each node transmits on a separate channel, causing no inter-node interference. Hence, any subset of nodes can transmit at the same slot, and the conflict graph does not contain any edges. For another example, consider a wireless network subject to

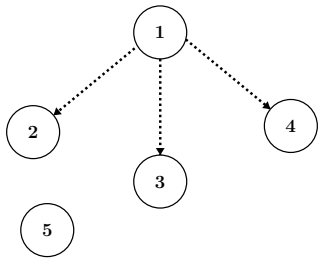


Figure 6-1: An example of packet transmission over hyperedges - when the node 1 transmits a packet, assuming no interference, it is received simultaneously by the neighboring nodes $2, 3$ and 4.

primary interference constraints. In this case, the edge (i, j) is *absent* in the conflict graph $\mathcal{C}(\mathcal{G})$ if and only if nodes i and j are *not* in the transmission range of each other and their out-neighbor-sets are disjoint. The set of all feasible transmission schedules $\mathcal{M}$ consists of the set of all *Independent Sets* in the conflict graph.

Note that the above definition of feasible schedules and conflict graph, does not allow *any* collision in the network. The same assumption was also used in [21], where such schedules were called "interference-free". However, due to the point-to-multi-point nature of the wireless medium, it is possible (and sometimes benefitial) to consider schedules that allow some collisions, so that a transmitted packet may be correctly received only by a strict subset of neighbors. As it will be clear in what follows, it is straightforward to extend our algorithm to allow such general schedules, albeit at the expense of additional computational complexity. In order to present the main ideas in a simplified setting, in the following, we stick to the "interference-free" schedules, as defined above.

6.2.3 The Broadcast Policy-Space Π

We first recall the definition of a connected dominating set of a graph $\mathcal{G}$ [4].

Definition 6.2.1 (Connected Dominating Set) *A connected dominating set D of a graph $\mathcal{G}(V, E)$ is a subset of vertices with the following properties:*

- *The source node* $\mathtt{r}$ *is in* D.
- *The induced subgraph* $\mathcal{G}(D)$ *is connected.*
- *Every vertex in the graph either belongs to the set* D *or is adjacent to a vertex in the set* D.

A connected dominating set D is called **minimal** if $D \setminus \{v\}$ is not a connected dominating set for any $v \in D$. The set of all minimal connected dominating set is denoted by $\mathcal{D}$.

A packet p is said to have been *broadcasted* by time t if the packet p is present at every node in the network by time t.

It is evident that a packet p is broadcasted if it has been transmitted sequentially by every node in a connected dominating set D. An admissible broadcast policy π is a sequence of actions $\{\pi_t\}_{t\geq 0}$ executed at every slot t. The action at time slot t consists of the following three operations:

1. **Route Selection:** Assign a connected dominating set $D \in \mathcal{D}$ to every incoming packet at the source $\mathtt{r}$ for routing.
2. **Node Activation:** Activate a subset of nodes from the set of all feasible activations $\mathcal{M}$.
3. **Packet Scheduling:** Transmit packets from the activated nodes according to some scheduling policy.

The set of all admissible broadcast policies is denoted by Π. The actions executed at every slot may depend on any past or future packet arrival and control actions.

Assume that under the action of the broadcast-policy π, the set of packets received by node i at the end of slot T is $N_i^\pi(T)$. Then the set of packets $B(T)$ received by all nodes, at the end of time T is given by

$$B^\pi(T) = \bigcap_{i\in V} N_i^\pi(T). \tag{6.1}$$

6.2.4 Broadcast Capacity λ^*

Let $R^\pi(T) = |B^\pi(T)|$ denote the number of packets delivered to all nodes in the network up to time T, under the action of an admissible policy π. Also assume that the external packets arrive at the source node with expected rate of λ packets per slot. The policy π is called a *broadcast policy of rate* λ if

$$\lim_{T\to\infty} \frac{R^\pi(T)}{T} = \lambda, \text{ w.p.1} \tag{6.2}$$

The broadcast capacity λ^* of the network is defined as

$$\lambda^* = \sup_{\pi\in\Pi}\{\lambda : \pi \text{ is a broadcast policy of rate } \lambda\} \tag{6.3}$$

The Wireless Broadcast problem is defined as finding an admissible policy π that achieves the Broadcast rate λ^*.

6.3 Hardness Results

Since a broadcast policy, as defined above, continues to be executed forever (compared to the finite termination property of standard algorithms), the usual notions of computational complexity theory do not directly apply in characterizing the complexity of these policies. Nevertheless, we show that the closely related problem of finite horizon broadcasting is **NP**-hard. Remarkably, this problem remains **NP**-hard even if the node activation constraints are relaxed (i.e., all nodes can transmit packets at the same slot, which is valid *e.g.*, when each node transmits over a different channel). Thus, the hardness of the problem arises from the combinatorial nature of distributing the packets among the nodes. This is in sharp contrast with the polynomially solvable Wired Broadcast problem where the broadcast nature of the wireless medium is non-existent and different outgoing edges from a node can transmit different packets over wire or directional antenna [14] [46] [66].

Consider the following finite horizon problem called **Wireless Broadcast**, with

the input parameters $\mathcal{G}, P, T$.

- **INSTANCE:** A Graph $\mathcal{G}(V, E)$ with capacities C on the nodes. A set $\mathcal{P}$ of P packets, located initially at the source, and a time horizon of T slots.
- **QUESTION:** Is there a scheduling algorithm π which routes all of these P packets to all nodes in the network by time T, i.e. $B^{\pi}(T) = \mathcal{P}$?

We prove the following hardness result:

Theorem 6.3.1 Wireless Broadcast *is* ***NP****-complete.*

Proof of Theorem 6.3.1 is based on reduction from the the **NP**-complete problem Monotone Not All Equal 3-SAT [67] to the Wireless Broadcast problem. Due to space limitations, we provide the proof of the Theorem in Appendix 8.1 of the techreport [68].

Note that the problem for $T = 1$ is trivial as only the out-neighbors of the source receive $\min(C, P)$ packets at the end of the first slot. The problem becomes non-trivial for any $T \geq 2$. In our reduction, we show that the problem is hard even for $T = 2$. This reduction technique may be extended in a straightforward fashion to show that the problem remains **NP**-complete for any fixed $T \geq 2$.

The above hardness result is in sharp contrast with the efficient solvability of the broadcast problem in the setting of point-to-point channels. In wired networks, the broadcast capacity can be achieved by routing packets using maximal edge-disjoint spanning trees, which can be efficiently computed using Edmonds' algorithm [14]. In a recent series of papers [46] [69], we proposed efficient throughput-optimal algorithms for wireless DAG networks in the static and time-varying settings. In a follow-up paper [66], the above line of work was extended to networks with arbitrary topology. In contrast, Theorem 6.3.1 and its corollary (see Appendix 8.1 of [68]) establishes that achieving the broadcast capacity in a wireless network with broadcast channel is intractable even for a simple network topology, such as a DAG. Also notice that

this hardness result is inherently different from the hardness result of [70], where the difficulty stems from the hardness of max-weight node activations, which is an Independent Set problem. The above result should also be contrasted with the hardness of the minimum energy broadcast problem [71].

6.4 Throughput-Optimal Broadcast Policy for a Relaxed Network

In this section, we give a brief outline of the design of the proposed broadcast policy, which will be described in detail in the subsequent sections. As in Chapter 5, the proposed policy consists of two interdependent modules - a control policy for a *precedence-relaxed* virtual network described below, and a control policy for the actual physical network, described in Section 6.5. Although, from a practical point of view, we are ultimately interested in the optimal control policy for the physical network, as we will soon see, this control policy is intimately related to, and derived from the dynamics of the relaxed virtual network.

6.4.1 Virtual Network and Virtual Queues

In this section we define and analyze the dynamics of an auxiliary virtual queueing process $\{\tilde{\boldsymbol{Q}}(t)\}_{t\geq 0}$. Our throughput-optimal broadcast policy π^* will be described in terms of the virtual queues. We emphasize that virtual queues are not physical entities and they do not contain any physical packet. They are constructed solely for the purpose of designing a throughput-optimal policy for the physical network, which depends only on the value of the virtual queue lengths. More interestingly, the designed virtual queues correspond to a fairly natural *single-hop* relaxation of the *multi-hop* physical network, as detailed below.

A Precedence-relaxed System

Consider an incoming packet p arriving at the source, which is to be broadcasted through a sequence of transmissions by nodes in a connected dominating set $D_p \in \mathcal{D}$. Appropriate choice of the set D_p is a part of our policy and will be discussed shortly. In reality, the packet p cannot be transmitted by a non-source node $v \in D_p$ at time t if it has not already reached the node v by the time t. This causality constraint is known as the *precedence constraint* in the literature [56]. We obtain the virtual queue process $\tilde{\boldsymbol{Q}}(t)$ by relaxing the precedence constraint, i.e., in the virtual queuing system, the packet p is made available for transmission by all nodes in the set D_p when the packet first arrives at the source. See Figure 6-2 for an illustration.

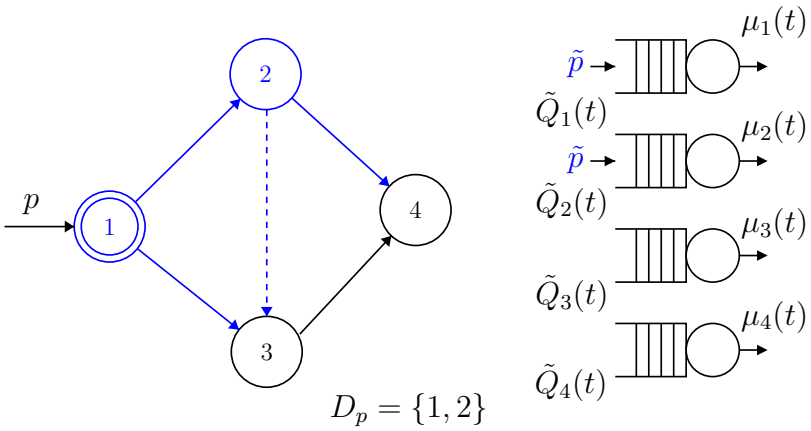


Figure 6-2: Illustration of the virtual queue system for the four-node wireless network $\mathcal{G}$. Upon arrival, the incoming packet p is prescribed a connected dominating set $D_p = \{1, 2\}$, which is used for its broadcasting. Relaxing the precedence constraint, packet p is counted as an arrival to the virtual queues $\tilde{Q}_1$ and $\tilde{Q}_2$ at the *same slot*. In the physical system, the packet p may take a while before reaching node 2, depending on the control policy.

Dynamics of the Virtual Queues

Formally, for each node $i \in V$, we define a virtual queue variable $\tilde{Q}_i(t)$. As described above, on the arrival of an external packet p at the source $\mathtt{r}$, the packet is replicated to a set of virtual queues $\{\tilde{Q}_i(t), i \in D_p\}$, where $D_p \in \mathcal{D}$ is a connected dominating set of the graph. Mathematically, this operation means that all virtual queue-counters in the set D_p are incremented by the number of external arrivals at the slot t. We will

use the control variable $A_i(t)$ to denote the number of packets that were routed to the virtual queue $\tilde{Q}_i$ at time t. The service rate $\boldsymbol{\mu}(t)$ allocated to the virtual queues is required to satisfy the same interference constraint as the physical network, i.e. $\boldsymbol{\mu}(t) \in \mathcal{M}, \forall t$. Hence, we can write the one step dynamics of the virtual queues as follows:

$$\tilde{Q}_i(t+1) = (\tilde{Q}_i(t) + A_i(t) - \mu_i(t))^+, \quad \forall i \in V \tag{6.4}$$

6.4.2 Dynamic Control of Virtual Queues

In this section, we design a dynamic control policy to stabilize the virtual queues for all arrival rates $\lambda < \lambda^*$. This policy takes action (choosing the routes of the incoming packets and selecting a feasible transmission schedule) by observing the virtual queue-lengths only and, unlike popular unicast policies such as *Backpressure*, does not require physical queue information. This control policy is obtained by minimizing one-step expected drift of an appropriately chosen Lyapunov function as described below. In the next section we will show how to combine this control policy for the virtual queues with an appropriate packet scheduling policy for the physical networks, so that the overall policy is throughput-optimal.

Consider the Lyapunov function $L(\cdot)$ defined as the Eucledian 2-norm of the virtual queue lengths, i.e.,

$$L(\tilde{\boldsymbol{Q}}(\boldsymbol{t})) = ||\tilde{\boldsymbol{Q}}(\boldsymbol{t})||_2 = \sqrt{\sum_i \tilde{Q}_i^2(t)} \tag{6.5}$$

The one step drift $\Delta(t)$ of the Lyapunov function may be bounded as follows:

$$\begin{aligned} \Delta(t) &\equiv L(\tilde{\boldsymbol{Q}}(\boldsymbol{t+1})) - L(\tilde{\boldsymbol{Q}}(\boldsymbol{t})) \\ &= \sqrt{\sum_i \tilde{Q}_i^2(t+1)} - \sqrt{\sum_i \tilde{Q}_i^2(t)} \end{aligned} \tag{6.6}$$

To bound this quantity, notice that for any $x \geq 0$ and $y > 0$, we have

$$\sqrt{x} - \sqrt{y} \leq \frac{x-y}{2\sqrt{y}} \tag{6.7}$$

The inequality above follows by noting that RHS minus LHS is non-negative. Substituting $x = ||\tilde{\boldsymbol{Q}}(t+1)||^2$ and $y = ||\tilde{\boldsymbol{Q}}(t)||^2$ in the inequality (6.7), we have the following bound on the one-step drift (6.6) for any $||\tilde{\boldsymbol{Q}}(t)|| > 0$

$$\Delta(t) \leq \frac{1}{2||\tilde{\boldsymbol{Q}}(t)||}\Big(\sum_i \big(\tilde{Q}_i^2(t+1) - \tilde{Q}_i^2(t)\big)\Big) \tag{6.8}$$

From the virtual queue dynamics (6.4), we have:

$$\begin{aligned}\tilde{Q}_i(t+1)^2 &\leq (\tilde{Q}_i(t) - \mu_i(t) + A_i(t))^2 \\ &= \tilde{Q}_i^2(t) + A_i^2(t) + \mu_i^2(t) + 2\tilde{Q}_i(t)A_i(t) - 2\tilde{Q}_i(t)\mu_i(t) - 2\mu_i(t)A_i(t)\end{aligned}$$

Since $\mu_i(t) \geq 0$ and $A_i(t) \geq 0$, we have

$$\tilde{Q}_i^2(t+1) - \tilde{Q}_i^2(t) \leq A_i^2(t) + \mu_i^2(t) + 2\tilde{Q}_i(t)A_i(t) - 2\tilde{Q}_i(t)\mu_i(t) \tag{6.9}$$

Hence, combining Eqns. (6.8) and (6.9), the one-step Lyapunov drift, conditional on the current virtual queue-length $\tilde{\boldsymbol{Q}}(t)$, under the action of an admissible policy π is upper-bounded as:

$$\begin{aligned}&\mathbb{E}(\Delta^{\pi}(t)|\tilde{\boldsymbol{Q}}(t) = \tilde{\boldsymbol{Q}}) \\ \overset{(\text{def})}{=}\ &\mathbb{E}\big(L(\tilde{\boldsymbol{Q}}(t+1)) - L(\tilde{\boldsymbol{Q}}(t))|\tilde{\boldsymbol{Q}}(t) = \tilde{\boldsymbol{Q}}\big) \\ \leq\ &\frac{1}{2||\tilde{\boldsymbol{Q}}||}\Big(B + 2\underbrace{\sum_i \tilde{Q}_i(t)\mathbb{E}\big(A_i^{\pi}(t)|\tilde{\boldsymbol{Q}}(t) = \tilde{\boldsymbol{Q}}\big)}_{(a)} \\ -\ &\underbrace{2\sum_i \tilde{Q}_i(t)\mathbb{E}\big(\mu_i^{\pi}(t)|\tilde{\boldsymbol{Q}}(t) = \tilde{\boldsymbol{Q}}\big)}_{(b)}\Big)\end{aligned} \tag{6.10}$$

where the constant $B = \sum_i (\mathbb{E}A_i^2(t) + \mathbb{E}\mu_i^2(t)) \leq n(\mathbb{E}A^2 + c_{\max}^2)$. By minimizing the upper-bound on drift from Eqn. (6.10), and exploiting the separable nature of the objective, we obtain the following control policy for the virtual queues:

Universal Max Weight (UMW) policy for the Virtual Queues

1. Route Selection: We minimize the term (a) in the above with respect to all feasible controls to obtain the following routing policy: Route the incoming packet at time t along the minimum-weight connected dominating set (MCDS) $D^{\mathrm{UMW}}(t)$, where the nodes are weighted by the virtual queue-lengths $\tilde{\boldsymbol{Q}}(t)$, i.e.,

$$D^{\mathrm{UMW}}(t) = \arg\min_{D \in \mathcal{D}} \sum_{i \in V} \tilde{Q}_i(t) \mathbb{1}(i \in \mathcal{D}) \tag{6.11}$$

2. Node Activations: We maximize the term (b) in the above with respect to all feasible controls to obtain the following node scheduling policy: At time t activate a feasible schedule $\boldsymbol{\mu}^{\mathrm{UMW}}(t)$ having the maximum weight, where the nodes are weighted by the virtual queue-lengths $\tilde{\boldsymbol{Q}}(t)$, i.e.,

$$M^{\mathrm{UMW}}(t) = \arg\max_{M \in \mathcal{M}} \sum_{i \in V} \tilde{Q}_i(t) c_i \mathbb{1}(i \in M) \tag{6.12}$$

In connection with the virtual queue systems $\tilde{\boldsymbol{Q}}(t)$, we establish the following theorem which will be essential in the proof of the throughput-optimality of the overall algorithm involving physical queues.

Theorem 6.4.1 *For any arrival rate $\lambda < \lambda^*$ the virtual queue process $\{\boldsymbol{Q}(t)\}_{t \geq 0}$ is positive recurrent under the action of the UMW policy and*

$$\max_i \tilde{Q}_i(t) = \mathcal{O}(\log t)^1, \quad w.p.\ 1.$$

The proof of Theorem 6.4.1 involves construction of an efficient randomized policy and using it with a sharper form of the Foster-Lyapunov theorem by Hajek [72]. This leads to the desired sample path result. The proof is provided in Appendix 6.8.2.

Discussion of the Result Even though the virtual queue process is positive recurrent under the action of the UMW policy, it is not true that they are uniformly bounded almost surely. Theorem 6.4.1 states that, instead, the virtual queue lengths increase at most logarithmically with time almost surely. Theorem 6.4.1 also strengthens the result of Theorem 2.8 of [33], where an almost sure $o(t)$ bound was established for the queue lengths[2].

In the rest of this chapter, we will primarily focus on the typical sample paths $\mathcal{E}$ of the virtual queue process satisfying the above almost sure bound. Formally, we define the set $\mathcal{E}$ to be

$$\max_i \tilde{Q}_i(\omega, t) = \mathcal{O}(\log(t)), \quad \forall \omega \in \mathcal{E}, \tag{6.13}$$

where $\mathbb{P}(\mathcal{E}) = 1$ from Theorem 6.4.1.

6.4.3 Bounds on the Virtual Queue

Recall that the random variable $A_i(t)$ denotes the total number of packets injected to the virtual queue $\tilde{Q}_i$ at time t. Similarly, the random variable $\mu_i(t)$ denotes the service rate from the virtual queue $\tilde{Q}_i$ at time t. Hence, the total number of packets that have been injected into any virtual queue $\tilde{Q}_i$ within the time interval $[t_1, t_2)$, $t_1 \leq t_2$ is given by

$$A_i(t_1, t_2) = \sum_{\tau=t_1}^{t_2-1} A_i(\tau). \tag{6.14}$$

[1]Recall that, $f(t) = \mathcal{O}(g(t))$ if there exist a positive constant c and a finite time t_0 such that $f(t) \leq cg(t), \forall t \geq t_0$.

[2]We say $f(t) = o(g(t))$ if $\lim_{t\to\infty} \frac{f(t)}{g(t)} = 0$.

Similarly, the total amount of service offered to the virtual queue $\tilde{Q}_i$ within the time interval $[t_1, t_2)$ is given by

$$S_i(t_1, t_2) = \sum_{\tau=t_1}^{t_2-1} \mu_i(\tau). \tag{6.15}$$

Using the well-known Skorokhod representation theorem [73] of the Queueing recursion (6.4), we have [3]

$$\tilde{Q}_i(t) = \sup_{1\leq\tau\leq t} \big(A_i(\tau, t) - S_i(\tau, t)\big)^+. \tag{6.16}$$

Since the virtual queues $\tilde{\boldsymbol{Q}}$ are controlled by the UMW policy, combining Eqn. (6.13) with (6.16), we have for all typical sample paths $\omega \in \mathcal{E}$:

$$A_i(\omega; \tau, t) \leq S_i(\omega; \tau, t) + F(\omega, t), \quad \forall \tau \leq t, i \in V, \tag{6.17}$$

where $F(\omega, t) = \mathcal{O}(\log t)$. In other words, equation (6.17) states that under the UMW policy, for any packet arrival rate $\lambda < \lambda^*$, the total number of packets that are routed to any virtual queue $\tilde{Q}_i$ may exceed the total amount of service offered to that queue in any time interval $[\tau, t)$ by at most an additive term of $\mathcal{O}(\log t)$ almost surely. In the following section, we will show that this arrival condition enables us to design a throughput-optimal broadcast policy.

6.5 Control of the Physical Network

With the help of the one-hop virtual queue structure designed in the previous section, we now focus our attention on designing a throughput-optimal control policy for the multi-hop physical network. Recall from Section 6.2 that a broadcast policy for the physical network is specified by the following three components: **(1)** Route Selection, **(2)** Node Activation, and **(3)** Packet Scheduling. In our proposed broadcast policy,

[3] Note that, for simplicity of notation and without any loss of generality, we have assumed the system to be empty at time $t = 0$.

components (1) and (2) for the physical network are identical to the corresponding components in the virtual network. In other words, an incoming packet p at time t is prescribed a route (i.e., a connected dominating set) given by Eqn. (6.11) and the set of nodes given by Eqn. (6.12) are scheduled for transmission in that slot. Note that, both these decisions are based on the instantaneous virtual queue lengths $\tilde{\boldsymbol{Q}}(t)$. In particular, it is possible that a particular node, with positive virtual queue length, is scheduled for transmission in a slot, even though it does not have any packets to transmit in its physical queue. The surprising fact, that will follow from Theorem 6.5.3, is that this kind of wasted transmissions are rare and *do not affect throughput.*

Packet Scheduling: There are many possibilities for the component **(3)**, i.e. Packet scheduling in the physical network. Recall that, the packet scheduling component selects packet(s) to be transmitted (subject to the node capacity constraint) when multiple packets contend for transmission by an active node and plays a role in determining the physical queuing process. In this chapter, we consider a priority based scheduler which gives priority to the packet which has been transmitted by the nodes *the least number of times.* We call this scheduling policy **L**east **T**ransmitted **F**irst or LTF. The LTF policy is inspired from the *Nearest To Origin* policy of Gamarnik [60], where it was shown to stabilize the queues for the unicast problem in wired networks in a deterministic adversarial setting. In spite of the high level similarities, however, we emphasize that these two policies are different, as the LTF policy works in the broadcast setting with point-to-multi-point transmissions and involves packet duplications.

Definition 6.5.1 (The policy LTF) *If multiple packets are available for transmission by an active node at the same time slot t, the LTF scheduling policy gives priority to a packet which has been transmitted the smallest number of times among all other contending packets.*

See Figure 6-3 for an illustration of the LTF policy.

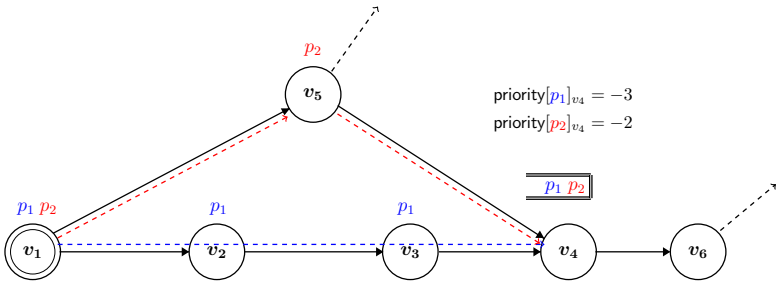


Figure 6-3: A schematic diagram depicting the scheduling policy LTF in action. The packet p_1's broadcast route consists of the nodes $\{v_1, v_2, v_3, v_4, \ldots\}$ and the packet p_2's broadcast route consists of the nodes $\{v_1, v_5, v_4, \ldots\}$ as shown in the figure. At node v_4, according to the LTF policy, the packet p_2 has higher priority than the packet p_1 for transmission.

6.5.1 Stability of the Physical Queues

Let us denote the length of the physical queue at node i at time t by $Q_i(t)$. Note that the number of packets which arrive at the source in the time interval $[\tau, t)$ and whose prescribed route contains the node i, is equal to the corresponding arrival in the virtual network $A_i(\tau, t)$, given by Eqn. (6.14). Similarly, total service offered by the physical node i in the time interval $(\tau, t]$ is given by $S_i(\tau, t)$, defined in Eqn. (6.15). Thus, the bound in Eqn. (6.17) may be interpreted in terms of the packets arriving to the physical network. This leads to the following theorem:

Theorem 6.5.2 *Under the action of the UMW policy with LTF packet scheduling, we have for any arrival rate $\lambda < \lambda^*$,*

$$\sum_{i \in V} Q_i(t) = \mathcal{O}(\log t), \quad \text{w.p.1.}$$

This implies that,

$$\lim_{t \to \infty} \frac{\sum_{i \in V} Q_i(t)}{t} = 0, \quad \text{w.p.1,}$$

i.e., the physical queues are rate-stable [33].

Theorem 6.5.2 is established by combining the sample path property of arrivals and departures from Eqn. (6.17), with an adversarial queueing theoretic argument [60]. Due to space limitations, we include the complete proof in Appendix 8.3 of the techreport [68].

As a direct consequence of Theorem 6.5.2, we have the main result of this chapter:

Theorem 6.5.3 *UMW is a throughput-optimal wireless broadcast policy.*

Proof The total number of packets $R(t)$, received by all nodes in common up to time t may be bounded in terms of the physical queue lengths as follows

$$A(0,t) - \sum_{i \in V} Q_i(t) \overset{(*)}{\leq} R(t) \leq A(0,t), \tag{6.18}$$

where the lower-bound $(*)$ follows from the simple observation that if a packet p has not reached at all nodes in the network, then at least one copy of it must be present in some physical queue.

Dividing both sides of Eqn. (6.18) by t, taking limits and using the Strong Law of Large Numbers and Theorem 6.5.2, we conclude that

$$\lim_{t \to \infty} \frac{R(t)}{t} = \lambda, \text{ w.p.1.}$$

Hence, from the definition (6.2.4), we conclude that UMW is throughput-optimal.

Efficient Implementation

We remind the reader that the routing and node activation decisions in UMW are made using the virtual queue lengths $\tilde{\boldsymbol{Q}}(t)$, whereas the physical packet scheduling decisions are based on the contents of the physical queues at each node. In the following, we discuss efficient implementation of each of the three components in detail.

Routing

A broadcast route (MCDS) is computed for each packet immediately upon its arrival according to Eqn. (6.11), and copied into its header field. The route selection involves solving an MCDS problem with the nodes weighted by the corresponding virtual queue lengths, which is **NP**-hard [74]. This is consistent with the hardness of the WIRELESS BROADCAST problem, established in Theorem 6.3.1. Assuming bi-directional wireless links, a polynomial time $\mathcal{O}(\log n)$ approximation algorithm for the MCDS problem is available for general graphs [75]. Furthermore, constant factor approximation algorithms for this problem are available for unit disk graphs [76].

Node Activation

At every slot a non-interfering subset of nodes is activated by choosing a maximum weight independent set in the conflict graph $\mathcal{C}(\mathcal{G})$, where the nodes are weighted by their corresponding virtual queue lengths, see Eqn. (6.12). The problem of finding a maximum weight independent set in a general graph is known to be **NP**-hard [74]. However, for the special case, such as unit disk graphs, constant factor approximation algorithms are available [77]. Note that, the same issue arises in the classical max-weight policies [1].

By a similar analysis, it can be shown that using an $\alpha \geq 1$ approximation algorithm for routing and $\beta \geq 1$ approximation algorithm for node activation, we can achieve $\frac{1}{\max(\alpha,\beta)}$ fraction of the optimal broadcast capacity of the network.

Packet Scheduling

The LTF policy can be efficiently implemented by maintaining a *min-heap* data-structure per node. The initial priority of each incoming packet at the source is set to zero. Once a packet p is received at a node i and the node i is included in its list of required transmitting node, its priority is decreased by one and it is inserted to the min-heap maintained at node i. Naturally, a node simply discards multiple receptions of the same packet.

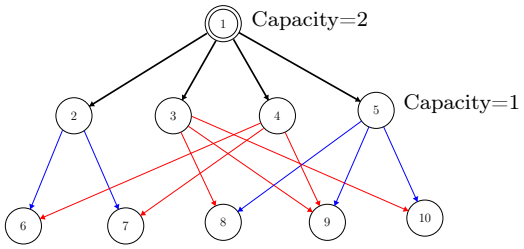


Figure 6-4: A wireless network with non-interfering channels. The broadcast capacity of the network is $\lambda^* = 2$.

6.6 Simulation Results

6.6.1 Interference-free Network

As a proof of concept, we first simulate the **UMW** policy in a simple wireless network with known broadcast capacity. Consider the network shown in Figure 6-4. Here node 1 is the source having a transmission capacity $C_1 = 2$. All other nodes in the network have unit transmission capacity. Assume that the channels are non-interfering, i.e., all nodes can transmit in a slot (this holds, e.g., if the nodes transmit on different frequencies). Since the broadcast capacity of any wireless network is upper-bounded by the capacity of the source, we readily have $\lambda^* \leq 2$. Also, it can be seen from Figure 6-4 that by transmitting the even numbered packets from nodes 2 and 5 (shown in blue) and the odd numbered packets from nodes 3 and 4, a broadcast rate of 2 packets per slot can be achieved. Hence, the broadcast capacity of the network is $\lambda^* = 2$. Figure 6-5 shows the average broadcast delay with the packet arrival rate λ in this network under the action of the proposed UMW policy. Note that the minimum delay is at least 2 as it takes at least two slots for any arriving packet to reach the nodes in the third layer. The plot confirms that the dynamic policy achieves the full broadcast capacity.

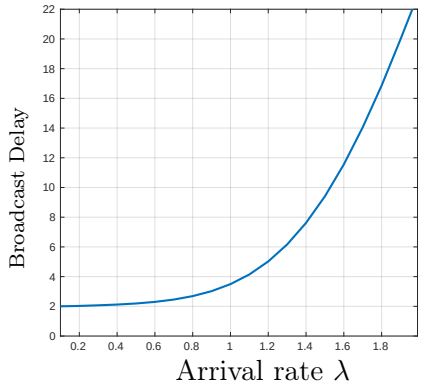


Figure 6-5: Plot of the broadcast delay incurred by the UMW policy as a function of the arrival rate λ in the network shown in Fig. 6-4.

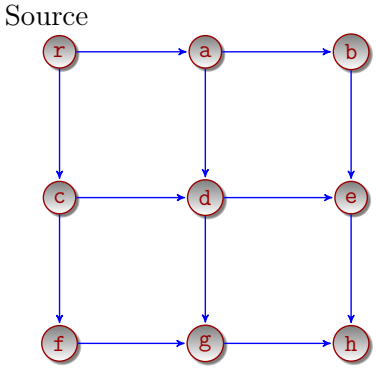


Figure 6-6: A 3×3 wireless grid network with primary interference constraints. The wireless broadcast capacity (λ^*) of the network is at most $\frac{1}{3}$.

6.6.2 Network with Interference Constraints

Consider the 3×3 wireless grid network, shown in Fig. 6-6. Assume that the transmissions are limited by primary interference constraints, i.e, two nodes cannot transmit together if the transmissions interfere at any node in the network. Assume that any node, if activated, has a transmission rate of one packet per slot. In this setting we have the following upper-bound on the broadcast capacity of the network.

Lemma 6.6.1 *The broadcast capacity of the* 3×3 *grid network is at most* $\frac{1}{3}$.

The proof of the lemma is provided in Appendix 8.4 of the techreport [68].

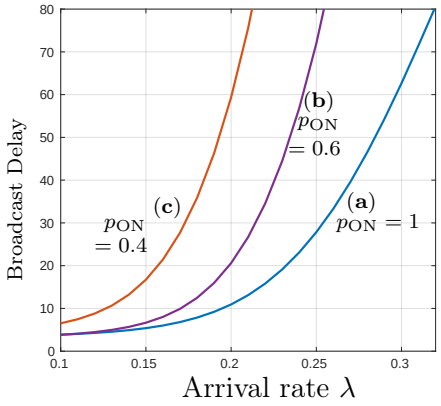


Figure 6-7: Plot of the broadcast delay incurred by the UMW policy as a function of the arrival rate λ in the 3×3 wireless grid network shown in Fig. 6-6.

In Figure 6-7 we show the broadcast delay as a function of the packet arrival rate, under the action of the UMW policy on the right most curve marked (a). From the plot, we observe that the delay-throughput curve has a vertical asymptote approximately along the straightline $\lambda = \frac{1}{3}$. This, together with lemma 6.6.1, immediately implies that the broadcast capacity of the network is $\lambda^* = \frac{1}{3}$ and confirms the throughput-optimality of the UMW policy.

Broadcasting in a Time Varying Network

Next, we simulate the UMW broadcast policy on a time-varying wireless grid network of Figure 6-6, in which the nodes are not always available for transmission (*e.g.,* they are sensors in sleep mode). In particular, we assume a simplified model where each node is active for potential transmission at a slot independently with some fixed but unknown probability p_{ON}. The delay performance of the proposed UMW broadcast policy is shown in Figure 6-7 (b) and (c) for two cases, $p_{\mathrm{ON}} = 0.6$ and $p_{\mathrm{ON}} = 0.4$ respectively. Following similar analysis as in the preceding sections, it can be shown that the UMW policy is also throughput-optimal for time-varying networks. Hence, from the plot it follows that the broadcast capacities of the time-varying 3×3 wireless grid network are ≈ 0.26 and ≈ 0.22 packets per slot, for the activity parameter $p_{\mathrm{ON}} = 0.6$ and $p_{\mathrm{ON}} = 0.4$ respectively.

6.7 Conclusion

In this chapter we obtained the first throughput-optimal broadcast policy for wireless networks with point-to-multi-point links and arbitrary scheduling constraints. The policy is derived using the powerful framework of *precedence-relaxed virtual network*, which we used earlier for designing throughput-optimal policies for networks with point-to-point links. Packet routing and scheduling decisions are made by solving standard optimization problems on the network, weighted by the virtual queue lengths. The policy is proved to be throughput optimal by a combination of Lyapunov method and a sample path argument using adversarial queueing theory. Extensive simulation results demonstrate the efficiency of the proposed policy in both static and dynamic network settings. There exist several interesting directions to extend this work. First, in our simplified model, we assumed that interference-free wireless transmissions are also error-free. A more accurate wireless channel model would incorporate the possibility of packet losses associated with each individual receiving nodes, due to fading and receiver noise [69]. Second, it remains unknown whether the UMW policy is still throughput optimal if the routing and node activations are made using the corresponding *physical queue* lengths as compared to the virtual queues. A positive result in this direction would lead to a more efficient implementation.

6.8 Appendix

6.8.1 Proof of Hardness of the Wireless Broadcast Problem

We start with the following lemma

Lemma 6.8.1 *Wireless Broadcast is in NP.*

Proof From the formulation, the problem Wireless Broadcast is a Decision Problem.

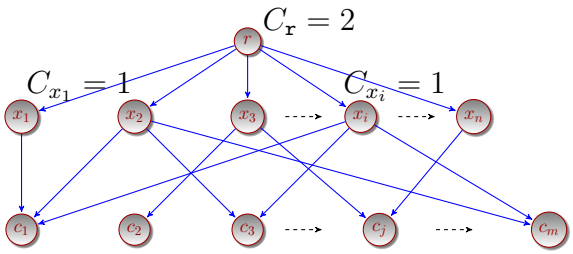


Figure 6-8: The Gadget used for the hardness proof

Also, if it has a Yes answer then there is a scheduling algorithm which serves as a certificate. Hence the problem belongs to NP.

Next we show that an **NP**-complete problem, named MONOTONE NOT ALL EQUAL 3-SAT (MNAE-3SAT) reduces to the problem WIRELESS BROADCAST in polynomial time. This will complete the reduction.
We begin with the description of the problem MNAE-3SAT:

The problem MNAE-3SAT

- **INSTANCE:** Set U of boolean variables, collection $\mathcal{C}$ of clauses over U such that each clause $c \in \mathcal{C}$ has $|c| = 3$ variables and none of the clauses contain complemented variables (*Monotonicity*).
- **QUESTION:** Is there a truth assignment for U such that each clause in $\mathcal{C}$ has at least one true literal and at least one false literal ?

It is known that the problem MNAE-3SAT is **NP**-complete [67].

Reduction: MNAE-3SAT $\overset{\text{poly}}{\Longrightarrow}$ WIRELESS BROADCAST

Suppose we are given an instance of the problem MNAE-3SAT $(U, \mathcal{C})$. Let $|U| = n$ and $|\mathcal{C}| = m$. Denote n boolean variables by $\{x_i, i = 1, 2, \ldots, n\}$. For this instance of MNAE-3SAT, we consider the following instance $\mathcal{G}(V, E)$ of WIRELESS BROADCAST as shown in Figure 6-8. The construction is done as follows:

- There are a total of $n + m + 1$ nodes. The nodes are divided into three layers as shown in Fig. 6-8.

- Let $\mathtt{r} \in V$ be the source node in the first layer. The capacity of the source node is 2. This means that, the source node can transmit 2 packets per slot to its out-neighbours.

- There are n nodes in the second layer of the figure 6-8, all of which are out-neighbors of the source node $\mathtt{r}$. Each of these nodes correspond to a variable x_i in MNAE-3SAT instance. Capacity of each of these nodes in the second layer is one.

- There are m nodes in the third layer, each corresponding to a clause $c \in \mathcal{C}$ in the MNAE-3SAT instance. The edges incoming to a node c_j are defined as follows: if the clause c_j is expressed as $c_j = x_{i_1} \vee x_{i_2} \vee x_{i_3}$, then we add three edges $(x_{i_1}, c_j), (x_{i_2}, c_j), (x_{i_3}, c_j)$ in the graph $\mathcal{G}(V, E)$. Capacities of each node in the third layer is taken to be 1.

Now consider the following instance of Wireless Broadcast on the constructed graph $\mathcal{G}(V, E)$. There are $M = 2$ packets at the source with a deadline of $T = 2$ slots. We claim that the following two questions are equivalent, i.e. Question 1 has a YES answer iff the Question 2 has an YES answer.

- **Question 1:** Is the MNAE-3SAT instance $(U, \mathcal{C})$ satisfiable ?

- **Question 2:** Does the constructed Wireless Broadcast instance have a Yes Solution?

To show this, let us denote the packets sent by the source $\mathtt{r}$ at the beginning of the slot by $\{0, 1\}$. Since the capacity of the source $\mathtt{r}$ is 2, all nodes $x_1, x_2, \ldots, x_n$ receive this packet at every slot. Since the capacities of the nodes x_i is only unity, they can only transmit either packet 0 or the packet 1 at that slot. We can denote this choice by the binary variable x_i, i.e. $x_i = 0$ if the node x_i sends packet 0 and is 1 if it sends

packet 1.

Note that the node c_j will receive both the packets if the corresponding clause contains at least one 0 and at least one 1. For a broadcast capacity of 2, all nodes must receive both packets at every slot. This is exactly the condition for the existence of a satisfying assignment of the MNAE-3SAT instance. This proves the intended hardness result.

■

Corollary:

As a direct consequence of the above reduction, it follows that the problem Wireless Broadcast remains **NP**-complete even with the following additional restrictions:

1. The wireless transmissions are non-interfering.
2. The graph $\mathcal{G}(V, E)$ is a DAG (c.f. [46], [66]).
3. The node capacities may take at most two values.
4. The in-degree of each node is at most 3.

6.8.2 Proof of Stability of the Virtual Queues

The proof of positive-recurrence and the sample path result is divided into several parts. First we describe and develop some general tools and then apply these tools to the virtual-queue Markov Chain $\{\tilde{\boldsymbol{Q}}(t)\}_{t\geq 1}$.

Mathematical Tools

The key to our proof is a stronger version of the Foster-Lyapunov drift theorem, obtained by Hajek [72] in a more general context. The following statement of the result, quoted from [78], will suffice our purpose. First, we recall the definition of a Lyapunov function:

Definition 6.8.2 (Lyapunov Function) *Let $\mathcal{X}$ denote the state space of any process. We call a function $L : \mathcal{X} \to \mathbb{R}$ a **Lyapunov** function if the following conditions hold:*

- *(1) $L(x) \geq 0, \forall x \in \mathcal{X}$ and,*
- *(2) the set $S(M) = \{x \in \mathcal{X} : L(x) \leq M\}$ is finite for all finite M.*

Theorem 6.8.3 (Hajek '82) *For an irreducible and aperiodic Markov Chain $\{X(t)\}_{t\geq 0}$ over a countable state space $\mathcal{X}$, suppose $L : \mathcal{X} \to \mathbb{R}_+$ is a Lyapunov function. Define the drift of L at X as*

$$\Delta L(X) \stackrel{\Delta}{=} \big(L(X(t+1)) - L(X(t))\big)\mathcal{I}(X(t) = X),$$

where $\mathcal{I}(\cdot)$ is the indicator function. Thus, $\Delta Z(X)$ is a random variable that measures the amount of change in the value of Z in one step, starting from the state X. Assume that the drift satisfies the following two conditions:

- ***(C1)** There exists an $\epsilon > 0$ and a $B < \infty$ such that*

$$\mathbb{E}(\Delta L(X)|X(t) = X) \leq -\epsilon, \ \forall X \in \mathcal{X} \text{ with } Z(X) \geq B$$

- ***(C2)** There exists a $D < \infty$ such that*

$$|\Delta L(X)| \leq D, \ w.p.\,1, \ \forall X \in \mathcal{X}$$

Then, the Markov Chain $\{X(t)\}_{t\geq 0}$ is positive recurrent. Furthermore, there exists scalars $\theta^* > 0$ and a $C^* < \infty$ such that

$$\limsup_{t\to\infty} \mathbb{E}\big(\exp(\theta^* L(X(t))\big) \leq C^*$$

We now establish the following technical lemma, which will be useful later.

Lemma 6.8.4 *Let $\{Y(t)\}_{t\geq 0}$ be a stochastic process taking values on the non-negative real line. Supppose that there exists scalars $\theta^* > 0$ and $C^* < \infty$ such that,*

$$\limsup_{t\to\infty} \mathbb{E}(\exp(\theta^* Y(t))) \leq C^* \tag{6.19}$$

Then,

$$Y(t) = \mathcal{O}(\log(t)), \quad w.p.1$$

Proof Define the positive constant $\eta^* = \frac{2}{\theta^*}$. We will show that

$$\mathbb{P}(Y(t) \geq \eta^* \log(t), \quad \text{i.o.}^4) = 0.$$

For this, define the event E_t as

$$E_t = \{Y(t) \geq \eta^* \log(t)\} \tag{6.20}$$

From the given condition (6.19), we know that there exists a finite time t^* such that

$$\mathbb{E}(\exp(\theta^* Y(t))) \leq C^* + 1, \ \forall t \geq t^* \tag{6.21}$$

We can now upper-bound the probabilities of the events $E_t, t \geq t^*$ as follows

$$\begin{aligned}
\mathbb{P}(E_t) &= \mathbb{P}(Y(t) \geq \eta^* \log(t)) \\
&= \mathbb{P}\big(\exp(\theta^* Y(t)) \geq \exp(\theta^* \eta^* \log(t))\big) \\
&\overset{(a)}{\leq} \frac{\mathbb{E}(\exp(\theta^* Y(t)))}{t^2} \\
&\overset{(b)}{\leq} \frac{C^* + 1}{t^2}
\end{aligned}$$

The inequality (a) follows from the Markov inequality and the fact that $\theta^*\eta^* = 2$. The inequality (b) follows from Eqn. (6.21). Thus, we have

$$\begin{aligned}\sum_{t=1}^{\infty}\mathbb{P}(E_t) &= \sum_{t=1}^{t^*-1}\mathbb{P}(E_t) + \sum_{t=t^*}^{\infty}\mathbb{P}(E_t) \\ &\leq t^* + (C^*+1)\sum_{t=t^*}^{\infty}\frac{1}{t^2} \\ &\leq t^* + (C^*+1)\frac{\pi^2}{6} < \infty\end{aligned}$$

Finally, using the Borel Cantelli Lemma, we conclude that

$$\mathbb{P}(\limsup Y_t \geq \eta^* \log t) = \mathbb{P}(E_t \text{ i.o.}) = 0$$

This proves that $Y_t = \mathcal{O}(\log t)$, w.p.1.

Combining Theorem 6.8.3 with Lemma 6.8.4, we have the following corollary

Corollary 6.8.5 *Under the conditions (C1) and (C2) of Theorem 6.8.3, we have*

$$L(X(t)) = \mathcal{O}(\log t), \quad w.p.\ 1$$

Construction of a Stationary Randomized Policy for the Virtual Queues $\{\tilde{\boldsymbol{Q}}(t)\}_{t\geq 1}$

Let $\mathcal{D}$ denote the set of all Connected Dominating Sets (CDS) in the graph $\mathcal{G}$ containing the source `r`. Since the broadcast rate $\lambda < \lambda^*$ is achievable by a stationary randomized policy, there exists such a policy π^* which executes the following:

- There exist non-negative scalars $\{a_i^*, i = 1, 2, \ldots, |\mathcal{D}|\}$ with $\sum_i a_i^* = \lambda$, such that each new incoming packet is routed independently along a CDS $D_i \in \mathcal{D}$ with probability $\frac{a_i^*}{\lambda}, \forall i$. The packet routed along the CDS D_i corresponds to an

arrival to the virtual queues $\{Q_j, j \in D_i\}$.

As a result, packets arrive to the virtual queue Q_j i.i.d. at an expected rate of $\sum_{i:j\in D_i} a_j^*, \forall j$ per slot.

- A feasible schedule $\boldsymbol{s}_j \in \mathcal{M}$ is selected for transmission with probability p_j $j = 1, 2, \ldots, k$ i.i.d. at every slot. By Caratheodory's theorem, the value of k can be restricted to at most $n+1$. This results in the following expected service rate vector from the virtual queues:

$$\boldsymbol{\mu}^* = \sum_{j=1}^{n+1} p_j c_j \boldsymbol{s}_j,$$

Since each of the virtual queues must be stable under the action of the policy π^*, from the theory of the GI/GI/1 queues, we know that there exists an $\epsilon > 0$ such that

$$\mu_i^* - \sum_{j:i\in D_j} a_j^* \geq \epsilon, \quad \forall i \in V \tag{6.22}$$

Next, we will verify that the conditions **C1** and **C2** in Theorem 6.8.3 holds for the Markov Chain of the virtual queues $\{\tilde{\boldsymbol{Q}}(t)\}_{t\geq 1}$ under the action of the **UMW** policy, with the Lyapunov function $L(\tilde{\boldsymbol{Q}}(t)) = ||\tilde{\boldsymbol{Q}}(t)||$ at any arrival rate $\lambda < \lambda^*$.

Verification of Condition (C1)- Negative Expected Drift

From the definition of the policy **UMW**, it minimizes the RHS of the drift upper-bound (6.10) from the set of all feasible policies Π. Hence, we can upper-bound the conditional drift of the **UMW** policy by comparing it with the stationary policy π^* described in 6.8.2 as follows:

$$\begin{aligned} & \mathbb{E}(\Delta^{\mathbf{UMW}}(t)|\tilde{\boldsymbol{Q}}(t) = \tilde{\boldsymbol{Q}}) \\ \leq \; & \frac{1}{2||\tilde{\boldsymbol{Q}}||}\Bigg(B + 2\sum_{i\in V} \tilde{Q}_i(t)\mathbb{E}\big(A_i^{\mathrm{UMW}}(t)|\tilde{\boldsymbol{Q}}(t) = \tilde{\boldsymbol{Q}}\big) \end{aligned}$$

$$
\begin{aligned}
&- \quad 2\sum_{i\in V}\tilde{Q}_i(t)\mathbb{E}\big(\mu_i^{\text{UMW}}(t)|\tilde{\boldsymbol{Q}}(t)=\tilde{\boldsymbol{Q}}\big)\Big)\\
&\overset{(a)}{\leq} \quad \frac{1}{2||\tilde{\boldsymbol{Q}}||}\Big(B+2\sum_{i\in V}\tilde{Q}_i(t)\mathbb{E}\big(A_i^{\pi^*}(t)|\tilde{\boldsymbol{Q}}(t)=\tilde{\boldsymbol{Q}}\big)\\
&- \quad 2\sum_{i\in V}\tilde{Q}_i(t)\mathbb{E}\big(\mu_i^{\pi^*}(t)|\tilde{\boldsymbol{Q}}(t)=\tilde{\boldsymbol{Q}}\big)\Big)\\
&= \quad \frac{1}{2||\tilde{\boldsymbol{Q}}||}\Big(B-2\sum_{i\in V}\tilde{Q}_i(t)\big(\mathbb{E}\mu_i^*(t)-\mathbb{E}A_i^*(t)\big)\Big)\\
&= \quad \frac{1}{2||\tilde{\boldsymbol{Q}}||}\Big(B-2\sum_{i\in V}\tilde{Q}_i(t)\big(\mu_i^*-\sum_{j:i\in D_j}a_j^*\big)\Big)\\
&\overset{(b)}{\leq} \quad \frac{B}{2||\tilde{\boldsymbol{Q}}||}-\epsilon\frac{\sum_{i\in V}\tilde{Q}_i(t)}{||\tilde{\boldsymbol{Q}}||},
\end{aligned}
$$

where inequality (a) follows from the definition of the **UMW** policy and inequality (b) follows from the stability property of the randomized policy given in Eqn. (6.22). Since the virtual-queue lengths $\tilde{\boldsymbol{Q}}(t)$ is a non-negative vector, it is easy to see that (*e.g.* by squaring both sides)

$$\sum_{i\in V}\tilde{Q}_i(t)\geq\sqrt{\sum_i\tilde{Q}_i^2(t)}=||\tilde{\boldsymbol{Q}}||$$

Hence, from Eqn. (6.23) in the above chain of inequalities, we obtain

$$\mathbb{E}(\Delta^{\mathbf{UMW}}(t)|\tilde{\boldsymbol{Q}}(t)=\tilde{\boldsymbol{Q}})\leq\frac{B}{2||\tilde{\boldsymbol{Q}}||}-\epsilon \tag{6.23}$$

Thus,

$$\mathbb{E}(\Delta^{\mathbf{UMW}}(t)|\tilde{\boldsymbol{Q}}(t))\leq-\frac{\epsilon}{2},\qquad\forall||\tilde{\boldsymbol{Q}}||\geq B/\epsilon$$

This verfies the negative expected drift condition **C1** in Theorem 6.8.3.

Verification of Condition (C2)- Almost Surely Bounded Drift

To show that the magnitude of one-step drift $|\Delta L(\tilde{\boldsymbol{Q}})|$ is almost surely bounded, we compute

$$\begin{aligned} |\Delta L(\tilde{\boldsymbol{Q}}(t))| &= |L(\tilde{\boldsymbol{Q}}(t+1)) - L(\tilde{\boldsymbol{Q}}(t))| \\ &= \big|||\tilde{\boldsymbol{Q}}(t+1)|| - ||\tilde{\boldsymbol{Q}}(t))|||\big| \end{aligned}$$

Now, from the dynamics of the virtual queues (6.4), we have for any virtual queue $\tilde{Q}_i$:

$$|\tilde{Q}_i(t+1) - \tilde{Q}_i(t)| \leq |A_i(t) - \mu_i(t)|$$

Thus,

$$||\tilde{\boldsymbol{Q}}(t+1) - \tilde{\boldsymbol{Q}}(t)|| \leq ||\boldsymbol{A}(t) - \boldsymbol{\mu}(t)|| \leq \sqrt{n}(A_{\max} + c_{\max})$$

Hence, using the triangle inequality for the ℓ_2 norm, we obtain

$$|\Delta L(\tilde{\boldsymbol{Q}}(t))| = \big|||\tilde{\boldsymbol{Q}}(t+1)|| - ||\tilde{\boldsymbol{Q}}(t))|||\big| \leq \sqrt{n}(A_{\max} + c_{\max}),$$

which verifies the condition **C2** of Theorem 6.8.3.

Almost Sure Bound on Virtual Queue Lengths

Finally, we invoke Corollary 6.8.5 to conclude that

$$\limsup_t ||\tilde{\boldsymbol{Q}}(t)|| = \mathcal{O}(\log t), \quad \text{w.p.1}$$

This implies that,

$$\max_i \tilde{\boldsymbol{Q}}_i(t) = \mathcal{O}(\log t), \quad \text{w.p.1} \quad \blacksquare$$

6.8.3 Proof of Theorem 6.5.2

Throughout this proof, we will consider only the typical sample point $\omega \in \mathcal{E}$ defined in Eqn. (6.13). For the sake of notational simplicity, we will drop the argument ω for evaluating a random variable $\boldsymbol{X}$ at ω, i.e., the deterministic sample path $X(\omega, t), \omega \in \mathcal{E}$ will be simply denoted by $X(t)$. We now make a simple observation which will be useful in the proof of the theorem:

Lemma 6.8.6 *Consider a function $F : \mathbb{Z}_+ \to \mathbb{Z}_+$, where $\mathbb{Z}_+$ is the set of non-negative integers. Assume that $F(t) = \mathcal{O}(\log t)$. Define $M(t) = \sup_{0 \leq \tau \leq t} F(\tau)$. Then,*

1. *$M(t)$ is non-decreasing in t and $M(t) \geq F(t)$.*
2. *$M(t) = \mathcal{O}(\log t)$.*

Proof Clearly, $M(t) \sup_{0 \leq \tau \leq t} F(\tau) \geq F(t)$ and

$$M(t+1) = \sup_{0 \leq \tau \leq t+1} F(\tau) \geq \sup_{0 \leq \tau \leq t} F(\tau) = M(t).$$

To prove the second claim, let $t_{\max}(t) = \arg\max_{0 \leq \tau \leq t} F(\tau)$. Clearly, $t_{\max}(t) \leq t$. Hence, for large enough t, we have

$$M(t) = F(t_{\max}(t)) = \mathcal{O}(\log t_{\max}(t)) = \mathcal{O}(\log t).$$

As a consequence of Lemma 6.8.6 applied to Eqn. (6.17), we have almost surely

$$A_i(t_0, t) \leq S_i(t_0, t) + M(t), \quad \forall i \in V, \, \forall t_0 \leq t, \tag{6.24}$$

for some non-decreasing function $M(t) = \mathcal{O}(\log t)$. We now return to the proof of the main the main result, Theorem 6.5.2.

of the main result Our proof technique is inspired by an adversarial queueing theory argument, given in [60]. We remind the reader that we are analyzing a typical sample path satisfying Eqn. (6.24), which holds almost surely. In the following argument, each copy of a packet is counted separately.

Without any loss of generality, assume that we start from an empty network at time $t = 0$. Let $R_k(t)$ denote the total number of packets waiting to be transmitted further at time t, which have already been forwarded *exactly* k times by the time t. We call such packets "layer k" packets. As we have mentioned earlier, if a packet is duplicated multiple times along its assigned route D (which is a connected dominating set (or CDS, in short)), each copy of the packet is counted separately in the variable $R_k(t)$, *i.e.,*

$$R_k(t) = \sum_{D \in \mathcal{D}} \sum_{i \in D_k} R_{(i,D)}(t), \tag{6.25}$$

where the variable $R_{(i,D)}(t)$ denotes the number of packets following the CDS D, that are waiting to be transmitted by the node $i \in D$ at time t and D_k is the set of nodes in the CDS D, which are exactly k^{th} hop away from the source along the CDS D. We show by induction that $R_k(t)$ is *almost surely* bounded by a function, which is $\mathcal{O}(\log t)$.

Base Step $k = 0$: Consider the source node $i = \mathtt{r}$ and an arbitrary time t. Let $t_0 \leq t$ be the largest time at which no packets of layer 0 (packets which are present only at the source and have never been transmitted before) were waiting to be transmitted by the source. If no such time exists, set $t_0 = 0$. During the time interval $(t_0, t]$, as a consequence of the property in Eqn. (6.24) of the **UMW** policy, at most $S_{\mathtt{r}}(t_0, t) + M(t)$ external packets have arrived to the source $\mathtt{r}$ for broadcasting. Also, by the choice of the time t_0, the source node $\mathtt{r}$ was always having packets to transmit during the entire time interval $(t_0, t]$. Since LTF packet scheduling policy is followed in the physical network, layer 0 packets have priority over all other packets

(in fact, there is packet of other layers present *only* at the source, but this is not the case at other nodes which we will consider in the induction step). Hence, it follows that the total number of layer 0 packets at time t satisfies

$$\begin{aligned} R_0(t) = \sum_{D\in\mathcal{D}} \sum_{i\in D_0} R_{(i,D)}(t) &\leq S_{\mathtt{r}}(t_0,t) + M(t) - S_{\mathtt{r}}(t_0,t) \\ &\leq M(t) \end{aligned} \tag{6.26}$$

Define $B_0(t) \stackrel{\text{def}}{=} M(t)$. Since $M(t) = \mathcal{O}(\log t)$, we have $B_0(t) = \mathcal{O}(\log t)$. Note that, since $M(t)$ is non-decreasing from Lemma (6.8.6), so is $B_0(t)$.

Induction Step: As our inductive assumption, suppose that, for some non-decreasing functions $B_j(t) = \mathcal{O}(\log t), j = 0, 1, 2, \ldots, k-1$, we have $R_j(t) \leq B_j(t)$, for all time t. We next show that there exists a non-decreasing function $B_k(t) = \mathcal{O}(\log t)$ such that $R_k(t) \leq B_k(t)$ for all time t.

To prove the above assertion, fix a node i and an arbitrary time t. Let $t_0 \leq t$ denote the largest time before t, such that there were no layer k packets waiting to be transmitted by the node i. Set $t_0 = 0$ if no such time exists. Hence the node i was always having packets to transmit during the time interval $(t_0, t]$ (packets in layer k or lower). The layer k packets that wait to be transmitted by the node i at time t are composed only of a subset of packets which were in layers $0 \leq j \leq k-1$ at time t_0 or packets that arrived during the time interval $(t_0, t]$ and include the node i as *one of their k^{th} transmitter* along the route followed. By our induction assumption, the first group of packets has a size bounded by $\sum_{j=0}^{k-1} B_j(t_0) \leq \sum_{j=0}^{k-1} B_j(t)$, where we have used the fact (using our induction step) that the functions $B_j(\cdot)$'s are monotonically non-decreasing. The size of the second group of packets is given by $\sum_{D:i\in D_k} A_D(t_0, t)$. We next estimate the number of layer k packets that crossed the edge e during the time interval $(t_0, t]$. Since the LTF packet scheduling policy is used in the physical network, layer k packets were not processed only when there were packets in layers up to $k-1$ that included the node i in its routing CDS. The number of such packets is bounded by $\sum_{j=0}^{k-1} B_j(t_0) \leq \sum_{j=0}^{k-1} B_j(t)$, which denotes the total possible number

of packets in layers up to $k-1$ at time t_0, *plus* $\sum_{j=0}^{k-1}\sum_{D:i\in D_j} A_D(t_0,t)$, which is the number of new packets that arrived in the interval $(t_0,t]$ and includes the node i as a transmitter within their first $k-1$ hops. Thus, we conclude that at least

$$\max\left\{0, S_i(t_0,t) - \sum_{j=0}^{k-1} B_j(t) - \sum_{j=0}^{k-1}\sum_{D:i\in D_j} A_D(t_0,t)\right\} \tag{6.27}$$

packets of layer k have been transmitted by the node i during the time interval $(t_0,t]$. Hence, the total number of layer k packets present at node i at time t is given as

$$\begin{aligned}
\textstyle\sum_{D:i\in D_k} R_{(i,D)}(t) &\le \sum_{j=0}^{k-1} B_j(t) + \sum_{D:i\in D_k} A_D(t_0,t) \\
- &\quad \big(S_i(t_0,t) - \sum_{j=0}^{k-1} B_j(t) - \sum_{j=0}^{k-1}\sum_{D:i\in D_j} A_D(t_0,t)\big) \\
= &\quad 2\sum_{j=0}^{k-1} B_j(t) + \sum_{j=0}^{k}\sum_{D:i\in D_j} A_D(t_0,t) - S_i(t_0,t) \\
\overset{(a)}{\le} &\quad 2\sum_{j=0}^{k-1} B_j(t) + A_i(t_0,t) - S_i(t_0,t) \\
\overset{(b)}{\le} &\quad 2\sum_{j=0}^{k-1} B_j(t) + M(t),
\end{aligned}$$

where the inequality (a) follows from the fact that each packet gets routed to a node i for transmission only once and hence

$$A_i(t_0,t) = \sum_{j=0}^{n-1}\sum_{D:i\in D_j} A_D(t_0,t), \quad \forall i \in V.$$

The inequality (b) follows from the property of the typical sample paths, stated in Eqn. (6.24). Hence, the total number of layer k packets at time t is bounded as

$$R_k(t) = \sum_i \sum_{D:i\in D_k} R_{(i,D)}(t) \le 2n\textstyle\sum_{j=0}^{k-1} B_j(t) + nM(t)$$

Define $B_k(t)$ to be the RHS of the above equation, i.e.

$$B_k(t) \stackrel{(\text{def})}{=} 2n \sum_{j=0}^{k-1} B_j(t) + nM(t) \tag{6.28}$$

Using our induction assumption and Eqn. (6.28), we conclude that $B_k(t) = \mathcal{O}(\log t)$, and it is easily seen to be non-decreasing. This completes the proof of the induction step.

To conclude the proof of the theorem, observe that the sum of the lengths of the physical queues at time t may be alternatively written as

$$\sum_{i \in V} Q_i(t) = \sum_{k=1}^{n-1} R_k(t) \tag{6.29}$$

Since the previous inductive argument shows that for all k, we have $R_k(t) \leq B_k(t)$ where $B_k(t) = \mathcal{O}(\log t)$ *a.s.*, we have $\sum_{i \in V} Q_i(t) = \mathcal{O}(\log t)$, and hence

$$\lim_{t \to \infty} \frac{\sum_{i \in V} Q_i(t)}{t} = 0, \quad \text{w.p. 1.} \tag{6.30}$$

6.8.4 Proof of Lemma 6.6.1

Proof Observe that, due to the primary interference constraints, the nodes $\mathtt{r}, a$ and d can not be activated at the same slot. Consider any arbitrary policy, which activates the node i for a fraction $f_i, i \in V$ times. Hence, we have the constraint that

$$f_{\mathtt{r}} + f_a + f_c \leq 1 \tag{6.31}$$

On the other hand, if the policy π achieves a broadcast rate of λ, it must be that

$$\begin{aligned}
\lambda &\leq f_{\mathtt{r}}, \quad \text{(considering broadcast rate at node } a\text{)} \\
\lambda &\leq f_a, \quad \text{(considering broadcast rate at node } b\text{)} \\
\lambda &\leq f_c, \quad \text{(considering broadcast rate at node } f\text{)}.
\end{aligned}$$

Adding the above three equations, we have

$$3\lambda \leq f_{\mathtt{r}} + f_a + f_c \overset{(a)}{\leq} 1, \tag{6.32}$$

where the inequality (a) follows from the constraint (6.31). Since the policy π is assumed to be arbitrary, we conclude that the broadcast capacity of the 3×3 grid network is at most $\frac{1}{3}$.

Bibliography

[1] 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.

[2] S. Sarkar and L. Tassiulas, "A framework for routing and congestion control for multicast information flows," *Information Theory, IEEE Transactions on*, vol. 48, no. 10, pp. 2690–2708, 2002.

[3] C. Joo, X. Lin, and N. B. Shroff, "Greedy maximal matching: Performance limits for arbitrary network graphs under the node-exclusive interference model," *Automatic Control, IEEE Transactions on*, vol. 54, no. 12, pp. 2734–2744, 2009.

[4] D. B. West *et al.*, *Introduction to graph theory*. Prentice hall Upper Saddle River, 2001, vol. 2.

[5] R. Ahlswede, N. Cai, S.-Y. Li, and R. W. Yeung, "Network information flow," *Information Theory, IEEE Transactions on*, vol. 46, no. 4, 2000.

[6] A. Karam, L. Zhang, and A. Lakas, "An efficient broadcasting scheme in support of military ad hoc communications in battle field," in *Innovations in Information Technology (IIT), 2013 9th International Conference on*. IEEE, 2013, pp. 78–83.

[7] Livestream®. [Online]. Available: http://new.livestream.com/

[8] R. Chen, W.-L. Jin, and A. Regan, "Broadcasting safety information in vehicular networks: issues and approaches," *IEEE network*, vol. 24, no. 1, pp. 20–25, 2010.

[9] S. Jiang, L. Guo, and X. Zhang, "Lightflood: an efficient flooding scheme for file search in unstructured peer-to-peer systems," in *Parallel Processing, 2003. Proceedings. 2003 International Conference on*, Oct 2003, pp. 627–635.

[10] M. Dietzfelbinger, "Gossiping and broadcasting versus computing functions in networks," *Discrete Applied Mathematics*, vol. 137, no. 2, pp. 127–153, 2004.

[11] I. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "A survey on sensor networks," *Communications Magazine, IEEE*, vol. 40, no. 8, pp. 102–114, Aug 2002.

[12] Y. Sasson, D. Cavin, and A. Schiper, "Probabilistic broadcast for flooding in wireless mobile ad hoc networks," in *Wireless Communications and Networking, 2003. WCNC 2003. 2003 IEEE*, vol. 2. IEEE, 2003, pp. 1124–1130.

[13] Y.-C. Tseng, S.-Y. Ni, Y.-S. Chen, and J.-P. Sheu, "The broadcast storm problem in a mobile ad hoc network," *Wireless networks*, vol. 8, no. 2-3, pp. 153–167, 2002.

[14] R. Rustin, *Combinatorial Algorithms*. Algorithmics Press, 1973.

[15] A. Czumaj and W. Rytter, "Broadcasting algorithms in radio networks with unknown topology," in *Foundations of Computer Science, 2003. Proceedings. 44th Annual IEEE Symposium on*. IEEE, 2003, pp. 492–501.

[16] G. K. Wong, H. Liu, X. Chu, Y.-W. Leung, and C. Xie, "Efficient broadcasting in multi-hop wireless networks with a realistic physical layer," *Ad Hoc Networks*, vol. 11, no. 4, pp. 1305–1318, 2013.

[17] J. Widmer, C. Fragouli, and J.-Y. Le Boudec, "Low-complexity energy-efficient broadcasting in wireless ad-hoc networks using network coding," in *In Proceedings*, no. LCA-CONF-2005-016, 2005.

[18] E. Kranakis, D. Krizanc, and A. Pelc, "Fault-tolerant broadcasting in radio networks," *Journal of Algorithms*, vol. 39, no. 1, pp. 47–67, 2001.

[19] L. Massoulie, A. Twigg, C. Gkantsidis, and P. Rodriguez, "Randomized decentralized broadcasting algorithms," in *INFOCOM 2007. 26th IEEE International Conference on Computer Communications. IEEE.* IEEE, 2007, pp. 1073–1081.

[20] T. Cui, L. Chen, and T. Ho, "Distributed optimization in wireless networks using broadcast advantage," in *Decision and Control, 2007 46th IEEE Conference on.* IEEE, 2007, pp. 5839–5844.

[21] D. Towsley and A. Twigg, "Rate-optimal decentralized broadcasting: the wireless case," in *ACITA*, 2008.

[22] K. Jain, M. Mahdian, and M. R. Salavatipour, "Packing steiner trees," in *Proceedings of the fourteenth annual ACM-SIAM symposium on Discrete algorithms.* Society for Industrial and Applied Mathematics, 2003.

[23] L. Bui, R. Srikant, and A. Stolyar, "Optimal resource allocation for multicast sessions in multi-hop wireless networks," *Philosophical Transactions of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, vol. 366, no. 1872, pp. 2059–2074, 2008.

[24] V. Badarla and D. J. Leith, "On delay performance of throughput optimal back-pressure routing: Testbed results." ICCCN, 2011.

[25] L. Ying, S. Shakkottai, A. Reddy, and S. Liu, "On combining shortest-path and back-pressure routing over multihop wireless networks," *IEEE/ACM Transactions on Networking (TON)*, vol. 19, no. 3, 2011.

[26] L. Bui, R. Srikant, and A. Stolyar, "Novel architectures and algorithms for delay reduction in back-pressure scheduling and routing," in *IEEE INFOCOM 2009*, April 2009, pp. 2936–2940.

[27] Y. Cui, E. M. Yeh, and R. Liu, "Enhancing the delay performance of dynamic backpressure algorithms," *IEEE/ACM Transactions on Networking*, vol. 24, no. 2, pp. 954–967, April 2016.

[28] M. Zargham, A. Ribeiro, and A. Jadbabaie, "Accelerated backpressure algorithm," in *Global Communications Conference (GLOBECOM), 2013 IEEE*. IEEE, 2013, pp. 2269–2275.

[29] A. Sinha, P. Mani, J. Liu, A. Flavel, and D. A. Maltz, "Distributed load management in anycast-based CDNs," in *2015 53rd Annual Allerton Conference on Communication, Control, and Computing (Allerton)*.

[30] J. Nunez-Martinez, J. Mangues-Bafalluy, and J. Baranda, "Anycast backpressure routing: Scalable mobile backhaul for dense small cell deployments," *IEEE Communications Letters*, vol. 17, no. 12, pp. 2316–2319, December 2013.

[31] S. S. Kulkarni and V. Badarla, "On multipath routing algorithm for software defined networks," in *2014 IEEE International Conference on Advanced Networks and Telecommuncations Systems (ANTS)*, Dec 2014, pp. 1–6.

[32] J. Baranda, J. NÃžÃśez-MartÃŋnez, and J. Mangues-Bafalluy, "Applying backpressure to balance resource usage in software-defined wireless backhauls," in *2015 IEEE International Conference on Communication Workshop (ICCW)*, June 2015, pp. 31–36.

[33] 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.

[34] T. M. Cover and J. A. Thomas, *Elements of information theory*. John Wiley & Sons, 2012.

[35] A. Brzezinski, G. Zussman, and E. Modiano, "Distributed throughput maximization in wireless mesh networks via pre-partitioning," *IEEE/ACM Transactions on Networking (TON)*, vol. 16, no. 6, pp. 1406–1419, 2008.

[36] L. X. Bui, S. Sanghavi, and R. Srikant, "Distributed link scheduling with constant overhead," *Networking, IEEE/ACM Transactions on*, vol. 17, no. 5, 2009.

[37] D. Bertsimas and J. N. Tsitsiklis, *Introduction to linear optimization.* Athena Scientific Belmont, MA, 1997, vol. 6.

[38] A. Schrijver, *Combinatorial optimization: polyhedra and efficiency.* Springer Science & Business Media, 2003, vol. 24.

[39] J. Matoušek, *Lectures on discrete geometry.* Springer New York, 2002, vol. 108.

[40] S. Dasgupta, C. H. Papadimitriou, and U. Vazirani, *Algorithms.* McGraw-Hill, Inc., 2006.

[41] Y. Chu, S. Rao, S. Seshan, and H. Zhang, "Enabling conferencing applications on the internet using an overlay multicast architecture," *ACM SIGCOMM computer communication review*, vol. 31, no. 4, pp. 55–67, 2001.

[42] D. Tse and P. Viswanath, *Fundamentals of wireless communication.* Cambridge university press, 2005.

[43] A. Kamthe, M. A. Carreira-Perpiñán, and A. E. Cerpa, "Improving wireless link simulation using multilevel markov models," *ACM Trans. Sen. Netw.*, vol. 10, no. 1, pp. 17:1–17:28, Dec. 2013. [Online]. Available: http://doi.acm.org/10.1145/2529991

[44] P. Agrawal and N. Patwari, "Correlated link shadow fading in multi-hop wireless networks," *Wireless Communications, IEEE Transactions on*, vol. 8, no. 8, pp. 4024–4036, 2009.

[45] N. Patwari and P. Agrawal, "Effects of correlated shadowing: Connectivity, localization, and rf tomography," in *Information Processing in Sensor Networks, 2008. IPSN'08. International Conference on.* IEEE, 2008, pp. 82–93.

[46] A. Sinha, G. Paschos, C.-p. Li, and E. Modiano, "Throughput-optimal broadcast on directed acyclic graphs," in *Computer Communications (INFOCOM), 2015 IEEE Conference on.* IEEE, 2015, pp. 1248–1256.

[47] D. V. Lindley, "The theory of queues with a single server," in *Mathematical Proceedings of the Cambridge Philosophical Society*, vol. 48. Cambridge Univ Press, 1952, pp. 277–289.

[48] R. G. Gallager, *Discrete stochastic processes*. Springer Science & Business Media, 2012, vol. 321.

[49] T. H. Cormen, *Introduction to algorithms*. MIT press, 2009.

[50] R. Durrett, *Probability: theory and examples*. Cambridge university press, 2010.

[51] E. Wong and B. Hajek, *Stochastic processes in engineering systems*. Springer Science & Business Media, 2012.

[52] M. Penrose, *Random geometric graphs*. Oxford University Press, 2003, no. 5.

[53] L. Bui, R. Srikant, and A. Stolyar, "Novel architectures and algorithms for delay reduction in back-pressure scheduling and routing," in *INFOCOM 2009, IEEE*. IEEE, 2009, pp. 2936–2940.

[54] D. B. Johnson and D. A. Maltz, "Dynamic source routing in ad hoc wireless networks," in *Mobile computing*. Springer, 1996, pp. 153–181.

[55] M. J. Neely, "Energy optimal control for time-varying wireless networks," *Information Theory, IEEE Transactions on*, vol. 52, no. 7, pp. 2915–2934, 2006.

[56] J. K. Lenstra and A. Rinnooy Kan, "Complexity of scheduling under precedence constraints," *Operations Research*, vol. 26, no. 1, pp. 22–35, 1978.

[57] J. Byrka, F. Grandoni, T. Rothvoß, and L. Sanità, "An improved LP-based approximation for Steiner tree," in *Proceedings of the forty-second ACM symposium on Theory of computing*. ACM, 2010, pp. 583–592.

[58] S. Meyn, *Control techniques for complex networks*. Cambridge University Press, 2008.

[59] M. Andrews, B. Awerbuch, A. Fernández, T. Leighton, Z. Liu, and J. Kleinberg, "Universal-stability results and performance bounds for greedy contention-resolution protocols," *Journal of the ACM (JACM)*, vol. 48, no. 1, pp. 39–69, 2001.

[60] D. Gamarnik, "Stability of adaptive and non-adaptive packet routing policies in adversarial queueing networks," in *Proceedings of the thirty-first annual ACM symposium on Theory of computing*. ACM, 1999.

[61] R. Chandra, C. Fetzer, and K. Hogstedt, "A mesh-based robust topology discovery algorithm for hybrid wireless networks," in *Proceedings of AD-HOC Networks and Wireless*, 2002.

[62] A. Sinha, G. Paschos, and E. Modiano, "Throughput-optimal multi-hop broadcast algorithms," in *Proceedings of the 17th ACM International Symposium on Mobile Ad Hoc Networking and Computing*, ser. MobiHoc '16. New York, NY, USA: ACM, 2016, pp. 51–60. [Online]. Available: http://doi.acm.org/10.1145/2942358.2942390

[63] N. Glick, "Breaking records and breaking boards," *American Mathematical Monthly*, pp. 2–26, 1978.

[64] K. Jain, J. Padhye, V. N. Padmanabhan, and L. Qiu, "Impact of interference on multi-hop wireless network performance," *Wireless networks*, vol. 11, no. 4, pp. 471–487, 2005.

[65] H. Li, Y. Cheng, C. Zhou, and P. Wan, "Multi-dimensional conflict graph based computing for optimal capacity in mr-mc wireless networks," in *2010 IEEE 30th International Conference on Distributed Computing Systems*, June 2010, pp. 774–783.

[66] A. Sinha, G. Paschos, and E. Modiano, "Throughput-optimal multi-hop broadcast algorithms," in *Mobihoc 2016*, no. Seventeenth International Symposium. ACM, 2016.

[67] T. J. Schaefer, "The complexity of satisfiability problems," in *Proceedings of the tenth annual ACM symposium on Theory of computing*. ACM, 1978, pp. 216–226.

[68] "Optimal Control for Generalized Network-Flow Problems," https://www.dropbox.com/s/1uf6aszpgmm1zvr/broadcast.pdf, Tech. Rep.

[69] A. Sinha, L. Tassiulas, and E. Modiano, "Throughput-optimal broadcast in wireless networks with dynamic topology," in *Mobihoc 2016*, no. Seventeenth International Symposium. ACM, 2016.

[70] D. Shah, N. David, and J. N. Tsitsiklis, "Hardness of low delay network scheduling," *IEEE Transactions on Information Theory*, vol. 57, no. 12, pp. 7810–7817, 2011.

[71] A. E. Clementi, P. Crescenzi, P. Penna, G. Rossi, and P. Vocca, "On the complexity of computing minimum energy consumption broadcast subgraphs," in *Annual Symposium on Theoretical Aspects of Computer Science*. Springer, 2001, pp. 121–131.

[72] B. Hajek, "Hitting-time and occupation-time bounds implied by drift analysis with applications," *Advances in Applied probability*, pp. 502–525, 1982.

[73] L. Kleinrock, "Queueing systems, volume i: theory," 1975.

[74] R. G. Michael and S. J. David, "Computers and intractability: a guide to the theory of np-completeness," *WH Free. Co., San Fr*, 1979.

[75] S. Guha and S. Khuller, "Approximation algorithms for connected dominating sets," *Algorithmica*, vol. 20, no. 4, pp. 374–387, 1998.

[76] M. T. Thai and D.-Z. Du, "Connected dominating sets in disk graphs with bidirectional links," *IEEE Communications Letters*, vol. 10, no. 3, pp. 138–140, 2006.

[77] G. K. Das, M. De, S. Kolay, S. C. Nandy, and S. Sur-Kolay, "Approximation algorithms for maximum independent set of a unit disk graph," *Information Processing Letters*, vol. 115, no. 3, pp. 439–446, 2015.

[78] A. Eryilmaz and R. Srikant, "Asymptotically tight steady-state queue length bounds implied by drift conditions," *Queueing Systems*, vol. 72, no. 3-4, pp. 311–359, 2012.