Backpressure Policies for Wireless ad hoc Networks

Umesh Kumar Shukla

Thesis submitted to the Faculty of the
Virginia Polytechnic Institute and State University
in partial fulfillment of the requirements for the degree of

Master of Science
in
Electrical Engineering

Allen B. MacKenzie, Chair
Luiz A. DaSilva
R. Michael Buehrer

March 16, 2010
Blacksburg, Virginia

Keywords: Wireless networks, Scheduling, Routing, Backpressure, Interference models

Backpressure Policies for Wireless ad hoc Networks

Umesh Kumar Shukla

(ABSTRACT)

Interference in ad hoc wireless networks causes the performance of traditional networking protocols to suffer. However, some user applications in ad hoc networks demand high throughput and low end-user delay. In the literature, the backpressure policy, i.e. queue backlog differential-based joint routing and scheduling, is known to be throughput-optimal with robust support for traffic load fluctuations [1]. Unfortunately, many backpressure-based algorithms cannot be implemented due to high end-user delay, inaccurate assumptions for interference, and high control overhead in distributed scenarios. We develop new backpressure based approaches to address these issues. We first propose a heuristic packet forwarding scheme that solves the issue of high end-user delay and still provides near-optimal throughput. Next we develop a novel interference model that provides simple yet accurate interference relationships among users. Such a model is helpful in designing a simple backpressure scheduling algorithm that does not violate realistic interference constraints. Finally we develop distributed backpressure algorithms based on our proposed ideas. Our distributed algorithms provide throughput performance close to the optimal and have low control overhead and simple implementation.

To mom, who always inspires me to stay positive

Acknowledgments

This work would not have been possible without my advisor, Dr. Allen MacKenzie who not only encouraged me to give my best to research but also motivated me to excel at academics. He gave me complete freedom to choose my own research problems. Throughout the past two years, he has taught me how to think of intuitive and simple solutions to complex problems. Professor Allen Mackenzie always set high standards for the work which has helped me shape up as a quality-driven professional. The education I have received under him is going to assist me for the time to come.

I am grateful to my committee members. Dr. Mike Buehrer had kindly agreed to supervise me during my first semester. It was in his classes where I got an opportunity learnt a lot about wireless systems. I am indebted to Dr. Luiz DaSilva who closely involved himself in my research. His results-driven approach and straightforward feedback always helped me keep the focus and target perfection.

I feel fortunate to have met sharp minds and turn them into lifelong friends, during my research experience at Virginia Tech. I thank Ryan Irwin who was always there to help me out and keep up my enthusiasm level. We enjoyed the time with chicken wings together. I also thank Haris Volos who has earned my tremendous respect for teaching me UWB measurements. Research could not have been fun without Abdallah, Amaar, Amr, Juan, Kaveh, Mustafa, Romi, and Sam, I could always count on these people for thoughtful suggestions and constructive inputs.

My stay at Blacksburg, a small college-town area was made adventurous by Ramakant

and Ramya who have together given me many happy moments. I thank my roommates Ajeet, Bharathram, and Harpreet who were seamlessly cooperative and always cheered me up. I definitely owe my good health while at Blacksburg to Barathram, Isha (Ma'am) and Shubhangi who never let me miss my home much - thanks for the great home-cooked indian food too.

Your family is the source of your strength and energy in all times. My mother has taught me to stay happy and optimistic, irrespective of any situation. For the immense love she has bestowed upon me, it is impossible to thank her in words. The constant support and encouragement of my sisters always helped me to chase my dreams. Last but not the least, Arpana, my best friend, maintained faith in my abilities throughout my ups and downs and always cheered me up. I am fortunate to have her with me for all coming years.

Contents

List of Figures

Chapter 1

Introduction

1.1 Motivation

With the popularity of wireless networks and the introduction of faster processors, new user applications require higher data rate and low end-user delay. The next-generation wireless networks must support these requirements as multimedia applications become indispensable. Many existing wireless protocols are designed for infrastructure based networks, and their utility in ad hoc networks without infrastructure is questionable. Ad hoc wireless networks are the most common example of infrastructure-less networks. In ad hoc wireless networks, a source node relies on the help of intermediate nodes to forward its traffic to the destination. The nodes in a wireless ad hoc network share the same wireless medium. If two transmitting nodes are geographically close and on the same channel, the transmission from one node could interfere with the other node's transmission. In this case, only one of them can transmit at a time. If every node makes its medium access decision independently, then interference will occur. If there are many collisions in the network, a packet may take a long time to be delivered to the destination, resulting in low throughput. Thus the protocols supporting high data rate applications in infrastructure-based networks may fail to support the same applications in ad hoc networks. However, ad hoc wireless networks are

needed in critical applications such as disasters where installing infrastructure is not feasible. These applications may require voice conversations, image sharing and real-time video. Such real-time applications require high data rate and low end-user delay. Thus we need better protocols for ad hoc wireless networks. These new network protocols should have the following basic features:

1. The network protocols should help users achieve maximum throughput.
2. The protocols should maintain end-to-end delay within tolerable limits. Note that maximizing throughput does not imply minimizing the end-to-end delay.
3. The protocols should make network decisions based on realistic interference conditions because interference is the main performance limiting factor in ad hoc networks.
4. The protocols should be distributed in order to be implemented in ad hoc networks.
5. The protocols should be "traffic aware" and robust to traffic fluctuations to accommodate the demands of every user.
6. The protocols should be simple to implement in real systems.
7. The protocols should be dynamic and scalable so that they can be implemented in mobile and large-sized networks.

Feature 2 suggests that such protocols must affect the network and transport layer design, while feature 3 requires the protocol to affect the link layer. Thus we require network routing and MAC protocols whose objective is to maximize the network throughput while minimizing delay. A class of network protocols, commonly known as backpressure-based protocols match with our basic criteria. The backpressure-based protocols are known to be throughput-optimal and do MAC-based scheduling and network routing jointly. Unfortunately, backpressure-based protocols do not satisfy all the criteria completely and have their own design challenges. Despite these shortcomings, backpressure-based protocols are

the best example of protocols that come close to meeting our objective. Our main goal is to get basic ideas from the backpressure protcols and develop a prototype algorithm that overcomes some of the shortcomings.

1.2 Background on Backpressure-based Protocols

A network policy determines medium access control and the traffic forwarding decisions in order to accomplish certain objectives. The capacity region of the network is defined as the set of all end-to-end traffic load matrices that can be stably supported under some network control policy. The stability means that all queues in the network have finite backlog. The capacity region of the network should be distinguished from the capacity region of a specific policy. The latter being the collection of all traffic load matrices that are sustainable by the specific policy. The larger the capacity region the better the performance will be, since the network will be stable for a wider range of traffic loads. The capacity region of the network is the union of the individual policy capacity regions over all possible control policies. A network policy is called *throughput-optimal* if its capacity region coincides with the network capacity region. In other words, a througput-optimal policy can stably support every end-to-end traffic load matrix in the network capacity region. The throughput would never increase if a node injects traffic at a rate that is outside the capacity region. In that case, the queues at intermediate nodes will overflow and the packets will be dropped before they reach the destination. As we will see later in this section, implementing a throughput-optimal policy may not be feasible in real systems. Also, the end-to-end delay might become high when a throughput-optimal policy supports a traffic load matrix close to the boundary of capacity region [2]. That is known as throughput-delay trade-off in wireless networks. Due to these constraints, it might be advisable to settle on a sub-optimal policy with a throughput region smaller than the network capacity region but that has tolerable end-to-end delay. However, the selected sub-optimal policy may be implemented in practice and may have lower end-to-end delay.

Before we discuss the details of backpressure policy, i.e. a joint routing and scheduling, let us briefly describe scheduling and routing and how to make them traffic aware to achieve higher throughput. Generally, a wireless medium is shared by multiple users whose simultaneous transmissions may interfere with each other. High interference among users can severely affect the system throughput. Thus it is important to decide which users should be allowed to transmit simultaneously in the network. A scheduling policy takes this decision. Its design is critical in achieving higher network throughput. The users or nodes with higher traffic should be given more chances to transmit. If all users were given equal chance to transmit, a user with negligible load may interfere with a heavily loaded user, causing resources to be wasted the throughput to suffer. A scheduling policy must account for the traffic loads to achieve higher throughput.

Routing between two nodes that are not in the transmission range of each other brings another set of challenges. Classically, a single shortest route between two users is selected to deliver the traffic in minimum amount of time. This strategy works if the traffic load is low and only a few flows are present in the network. But if a large number of flows exist and the traffic load is high, the shortest paths might be over-utilized and congested. Then queues of the intermediate nodes on the shortest path may build up quickly and the queuing delay may become high, increasing end-to-end delay. In those scenarios, it might be a good idea to explore longer but less congested routes to speed up packet-delivery and increase throughput.

Next we introduce the backpressure policy as originally presented in [1,3]. These policies are later generalized for mobile ad hoc wireless networks [2,4–6]. A backpressure policy gives forwarding priority to the traffic with higher backlog differentials. It discourages transmitting to congested nodes. This effectively sends the congestion notification to intermediate nodes and ultimately to the source. In that manner the source and intermediate nodes can make better forwarding and scheduling decisions. The backpressure policy consists of the following components: Define V as a set of nodes forming the network. Every node $a \in V$ has separate queue Q_a^c for each destination $c \in V \setminus \{a\}$ that it is serving. The notation $Q_a^c(t)$

indicates the length of queue Q_a^c at time t. A *neighbor* of node a is another node that can be reached in a single hop.

1. **Destination Selection:** At time t, node a first calculates the maximum difference of queue backlogs Q_{ab}^* for each of its neighbors b as follows

$$\begin{aligned} c_{ab}^*(t) &= \arg\max_{c \in V \setminus \{a\}} \{Q_a^c(t) - Q_b^c(t)\} \\ Q_{ab}^*(t) &= \max\left\{Q_a^{c_{ab}^*(t)}(t) - Q_b^{c_{ab}^*(t)}(t), 0\right\} \end{aligned} \tag{1.1}$$

   Node a selects the destination $c_{ab}^*(t)$ for forwarding on link (a, b) if link (a, b) is activated in step 2.

2. **Scheduling:** Define a binary activation variable x_{ab} for link (a, b) as follows

$$x_{ab}(t) = \begin{cases} 1 & \text{if link } (a,b) \text{ is scheduled at time } t \\ 0 & \text{Otherwise} \end{cases}$$

   Let X is a vector representing a combination of activation variables for all links and Π be a set containing all possible links schedules. Let $r_{ab}(X, t)$ be the link rate. Choose an optimum schedule $X^*(t)$ such that

$$X^*(t) = \arg\max_{X \in \Pi} \sum_{(a,b)} Q_{ab}^*(t) r_{ab}(X, t) x_{ab}(t) \tag{1.2}$$

   Note that link rate depends upon the interference experienced by that link, which is itself determined in part by which other links are scheduled in X(t). A backpressure scheduling algorithm determines the schedule based on the knowledge of queue backlog differential and the interference for every link in the network. These two inputs i.e. maximum queue backlog differential and the interference are the key inputs to backpressure scheduling algorithm.

3. **Traffic Forwarding/Routing:** If $x_{ab}(t) = 1$, node a will forward the traffic of destination $c_{ab}^*(t)$ to node b at time t. The traffic forwarding rate is $r_{ab}(X^*(t), t)$. Thus, the routes are determined on the basis of differential queue backlog.

The structure of backpressure policy exhibits a cross-layer design and requires interaction between MAC and network layer. All three components of backpressure have contribution in the achieving the network capacity. The routing and destination-selecton component are straightforward to realize in practice. However, the scheduling component is not easy to implement. The optimization problem in (1.2) has to consider a large number of feasible activation vectors; in fact, the number of feasible activation vectors is of exponential order with respect to number of nodes [1]. Thus, the optimization problem in (1.2) is intractable to solve online for large networks, although there are many existing approximation and heuristic algorithms in the literature. We will overview them in section 1.4.

1.3 Challenges for Backpressure Policies

As presented in the previous section, a backpressure policy makes packet forwarding decisions based on the differential queue backlog only. Backpressure routing utilizes all possible paths between source and destination to avoid congestion on any single path. Though this property is instrumental in achieving higher throughput when traffic load is high, backpressure routing will use longer routes and even looping routes when the traffic load is light. This leads to large unnecessary end-to-end packet delay. Moreover, using longer routes in such cases wastes network resources. Routing-loop formation is another drawback of backpressure routing. In many real time applications like voice and video, high end-to-end packet delay is unacceptable. Often in such applications, a packet received with high delay is no better than packet loss. We could prevent high end-to-end delay in backpressure routing by not forwarding the packets on longer paths. But we still want to maintain the sufficient routes for any source-destination pair to provide adequate load balancing in case of high traffic load. Generally, these two objectives conflict with each other because few short routes exist in the network.

A second problem in many backpressure schemes is the assumption of a simplistic interference model. As stated before, the scheduling algorithm heavily depends upon the interfer-

ence. Most past work exploits graph-theoretic techniques to design a scheduling algorithm. A major advantage of graph theoretic techniques is that a graph abstraction of the network is quite useful for higher layer protocol design. However, previous graph-theoretic scheduling approaches works adopt a disk-based interference model which assumes that interference ends abruptly at some boundary and does not propagate further. This model underestimates interference by ignoring the accumulative interference due to far away transmissions. Sometimes, it overestimates the interference as well by not allowing close-by links [7]. Thus the claimed performance of a scheduling policy using disk based interference model is questionable. The signal to interference and noise ratio (SINR) model is more accurate, but it is complex to design a scheduling policy for such model.

Implementing backpressure in a distributed way is another challenge. Backpressure scheduling is the main bottleneck as all nodes have to make the scheduling decisions jointly and they need to share a lot of information to reach a final schedule. Some distributed backpressure scheduling algorithms are proposed in the literature. Unfortunately, they are complex to implement and have high overhead. It is a big challenge to implement distributed backpressure scheduling which is simple and has low control overhead.

Backpressure routing utilizes multiple routes simultaneously to deliver packets to the destination. Due to this phenomenon, packets at the destination may be received out of sequence. Transport layer protocols like TCP originally designed for wired networks see this out-of-order delivery as a sign of congestion and ask the source node to perform flow control. This is known as the packet reordering problem. Thus if backpressure routing (or any load-balancing routing) is used along with TCP, we must ensure that packet reordering doesn't take place or mitigate TCP response to out-of-order packet delivery.

1.4 Related Work

Tssailus and Ephermides originally proposed the backpressure scheduling policies in [1, 3]. Since then the scheduling policies have been an active area of research. The original schedul-

ing policy schedules the maximum weighted independent set of all links in the network and is an NP-complete problem. Thus suboptimal but simpler designs of scheduling policies have been widely investigated. Among them, the most popular is a greedy scheduling policy which sequentially admits links to a schedule in a greedy fashion. The research in this area can be categorized in two broad classes: First, some research is focused on finding the guaranteed fraction of the capacity region that a scheduling algorithm can achieve. [8–11] proved that a greedy scheduling policy attains at least half of the capacity region. [12] showed that greedy scheduling attains the full capacity region in several different networks. [13–15] showed that greedy scheduling policies *empirically* perform nearly as well as an optimal scheduling policy. All these schemes assume an omniscent centralized controller which makes scheduling decisions. This is an unrealistic assumption in ad hoc networks where nodes make their network decisions autonomously. The second class of research in scheduling policies addresses this issue and presents designs of distributed scheduling algorithm. Distributed versions of greedy scheduling policies are presented in [16–19]. Tssailus in [20] provided randomized scheduling schemes that attain the maximum throughput region, which can be implemented in fully distributed manner using gossip-based algorithms [21]. Unfortunately, these schemes have large control overhead as all nodes require substantial amount of information exchange among each other. Recently [22] proposed a CSMA based distributed backpressure scheduling that requires no information exchange among nodes. However, the proposed algorithm utilizes learning-based mechanism to converge to an optimal schedule after reasonable amount of time. Under varying traffic conditions, such algorithm may not converge to a favorable schedule.

Neely presented a generalized version of backpressure policy which takes routing and scheduling decisions jointly [4, 23]. Though he presented the policy as a joint routing and power control technique, the power control technique can be seen as a generalized version of scheduling. He also proposed a backpressure policy which does joint rate control, routing, and scheduling [24]. There have been numerous versions of backpressure policies which do rate control, routing, scheduling, and power control jointly or in an independent man-

ner [25–30]. Some notable works that do not talk about backpressure specifically but have similar objectives and can be extended to backpressure policies [31–33]. Most of these techniques utilize convex optimization to find the resource allocation solutions. A good tutorial on these techniques can be found in [34].

After the idea of backpressure routing is proposed, the issue of high end-to-end delay is recognized and addressed in some later works. A few previous works deal with combining shortest path routing with backpressure routing [35, 36]. [35] presents a heuristic algorithm which uses a combined metric of differential queue backlog (to be defined later) and hop count to determine the next hop. [36] presents an algorithm which provably minimizes the average length of routes used for packet forwarding. Though both of these algorithms are throughput optimal, they may still choose longer paths to forward the packets when there is a traffic burst for a small time interval. Thus, their routing scheme may suffer from high delay in some realistic scenarios. Most works in backpressure policies so far rely on standard mathematical techniques in optimization and control systems and and are difficult to apply to practical systems directly. Recently, there have been some efforts to adapt backpressure policies for practical wireless networks. Some practical implementation of backpressure schemes (joint routing and rate control) have been proposed recently [37–39]. These works focus on achieving throughput-optimality; delay is not their main concern.

As discussed before, most works in scheduling policies assume a simplified disk-based model which is unrealistic. There are few works which consider scheduling in the SINR model. Some initial works adopt linear and non-linear programming approaches [40–42] that are not computationally efficient even for moderate-sized networks. Recently, there has been significant interest in the design of efficient scheduling policies satisfying SINR constraints [43–45]. The work in [44] assumes slow load variations in the network. Their policy learns the traffic patterns and then carries out optimal scheduling decisions every frame until the pattern changes again. But the traffic fluctuations may not be slow in some networks. Also, in case of arbitrary, bursty traffic arrivals, such scheduling policy may take a long time to converge. The authors in [45] present a frame-based heuristic scheduling policy over

an SINR-based interference graph which can be extended to slot based scheduling. In the interference graph, each node has a weighted edge to every other node in the network. A weight represents the amount of power one node receives from another. An approach similar to interference graph is presented in [42]. The scheduling in an interference graph requires global information to add or remove a single link from the schedule. Thus despite its features like accuracy, the interference graph approach is not attractive for practical implementation. A binary conflict graph is more expedient than an interference graph from a protocol design point of view as it paves the way to distributed design.

Finally, the issue of packet reordering caused by load-balancing routing, such as is utilized in backpressure, is addressed in a few past works. [46] provides a good tutorial on this issue. [47, 48] propose an interesting strategy which uses the bursty nature of TCP to avoid packet reordering. Some other approaches to address this problem can be found in [49–51].

1.5 Our Contribution

We examine three major issues in backpressure-based policies mentioned in section 1.3: High end-to-end delay in backpressure routing, realistic but graph-based interference models for backpressure scheduling, and the design of a simple and fully distributed backpressure policy. First, we propose a new packet forwarding strategy to address high end-to-end delay in backpressure routing. The proposed idea is to restrict the packet forwarding to shorter routes while still using the backpressure approach. We use a generic additive metric-based routing protocol for route discovery. In most cases, our packet forwarding strategy achieves a throughput close to original backpressure routing with good end-to-end delay performance. The end-to-end delay performance of our modified backpressure policy is very good. Using our strategy, a backpressure policy can achieve near-optimal throughput performance with fairly low end-to-end delay. Our strategy does not make assumptions regarding underlying routing protocol and any routing mechanism (distance vector, link state routing) can be combined with our scheme. Our approach is novel and simple to implement. We show

through simulation results that forwarding packets to longer routes is of limited benefit.

Next we examine the design of a realistic interference model that can take advantage of graph-theoretic scheduling techniques. We propose a novel interference model that facilitates simple scheduling policies while meeting realistic SINR constraints. Our main contribution is the design of a binary conflict graph that satisfies SINR constraints in both the uniform and non-uniform power cases. To the best of our knowledge, ours is the first effort of this kind, although we leverage the technique in [43] for the power control case. We perform backpressure scheduling over the conflict graph to maximize the network throughput while guaranteeing that the SINR of each scheduled link is greater than the required threshold. We simulate the performance of this backpressure scheduling policy over proposed conflict graphs and compare them with other popular interference models. The simulated performance of policies based on our conflict graph should match well with their performance in realistic scenarios.

Finally, we examine the design of distributed backpressure scheduling and combine it with our modified backpressure policy. Our distributed backpressure scheme performs quite well when compared to an optimal centralized policy. We also propose a distributed scheduling scheme with realistic interference constraints and analyze its performance. Our distributed backpressure policy has considerably less overhead as compared to existing policies and is simple to implement. We sketch possible extensions of our work to multichannel networks, e.g. cognitive radio networks, and present guidelines for a multichannel backpressure policy. While many resource allocation policies in the literature make assumptions on network topology and traffic statistics, our results hold true for arbitrary topologies and arbitrary traffic arrival statistics.

1.6 Outline

This thesis is divided into the following chapters: Chapter 2 discusses our backpressure-based packet forwarding scheme, which does significantly better in delay performance than

the original backpressure policy, and presents related simulation results. The backpressure-scheduling policy for the SINR model and related simulation results are presented in chapter 3. Chapter 4 presents the design and simulation results of a distributed backpressure policy. Chapter 5 presents guidelines for designing a backpressure policy for multichannel networks and concludes the thesis.

Chapter 2

Backpressure Based Packet Forwarding Scheme

As we discussed in the chapter 1, backpressure routing combined with backpressure scheduling is throughput-optimal in multi-hop ad hoc networks. Under backpressure routing scheme, a node forwards packets to those next-hop neighbors which have shorter queues. But a forwarding node does not care about how many extra hops may be required to deliver the packet to the destination. This is why, though this property helps avoid the congestion, it does not account for the delay accrued in forwarding packets on long routes when the traffic load is lower. This is shown in Figure 2.1. The end-to-end delay under light traffic load is unnecessarily high as compared to shortest path routing.

The performance of backpressure with low traffic makes it unattractive to implement in realistic wireless networks. We address this issue by making modifications to the original backpressure routing scheme. Minimum hop routing is a good choice with low traffic while backpressure routing should be chosen for high traffic scenarios. We combine both of them in an effort to leverage the benefits of both. In our modified scheme, we try to ensure that under light traffic, only shortest paths are utilized; as traffic load increases, we also start using longer paths. But we never use long paths as we will show in this chapter that

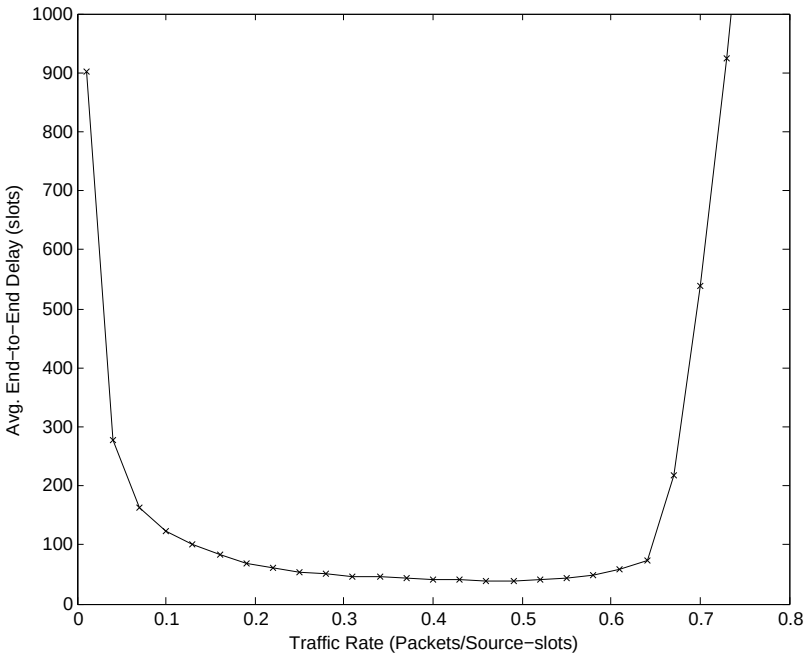

Figure 2.1: Delay performance of backpressure scheme under light traffic load

forwarding traffic on longer routes gives little extra gain. We will see later in the chapter that our modified scheme achieves network capacity quite close to what is achieved under original, throughput-optimal backpressure scheme. The rest of the chapter is organized as follows: In section 2.1, we formally describe our network model. We introduce our proposed schemes and underlying optimization framework in section 2.2. The simulation model and results are discussed in section 2.3, followed by the conclusion. Our work can also be found in [52].

2.1 Network Model

Define a graph $G = (V, E)$ representing the network. V is the set of nodes and E is the set of undirected links in the network. For the sake of simplicity, we consider the primary interference model where all disjoint links in the network can be activated simultaneously. This further implies that a node cannot transmit to or receive from more than one node at a time. This interference model is sometimes referred as node exclusive interference

model. Every node maintains an internal queue for every known destination in the network to facilitate backpressure-based routing. The complexity of this queuing architecture can be reduced by using suggestions given in [53]. Time is divided into equal slots. All links have equal rate of one packet per slot. Most of these assumptions are for the sake of clarity and ease of the simulation and are not restrictions on our proposed technique.

2.2 Backpressure Routing Schemes

As discussed before, an optimal scheduling algorithm is computationally intractable for large networks but its approximation exists. For example, it is empirically observed that greedy scheduling performs nearly as well as an optimal scheduling policy [14]. According to greedy backpressure scheduling scheme, we first activate a link with the highest value of maximum differential queue backlog and put it into a schedule. All other links conflicting to this link (which are all adjacent links under primary interference model) are not activated and are removed from the link set. We continue this iterative process until the link set is empty. Though greedy scheduling is centralized and impractical to implement in distributed systems, it serves our purpose well as the focus of our chapter is on backpressure routing, not scheduling. Though many solutions for distributed scheduling exist in the literature [16–19], they are beyond the scope of this chapter.

Our scheme uses a generic routing protocol for route discovery which can accommodate any additive routing metric. We modify original backpressure (Pure BP) approach and propose a scheme where a node forwards a packet to those next hops only which are either on the shortest paths or are equally far from the destination as the forwarding node. Let us denote later type of paths by *extended path* (EP). We call our scheme modified backpressure (Modified BP). For the sake of exposition, we will consider hop-count based routing, a special case of generic routing. Though the basic idea is to restrict packet forwarding to shorter routes, using the shortest routes only may not be sufficient. Often, the shortest paths for a given source-destination pair may be small in number, and the load balancing obtained

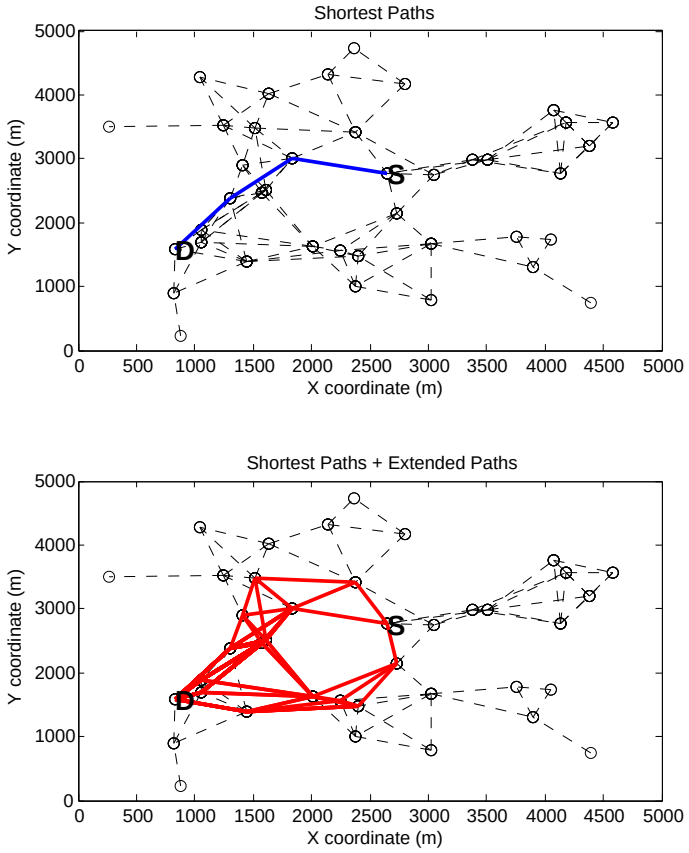

Figure 2.2: The shortest and extended paths in random topology

using the shortest paths might not be effective. But the number of extended paths might be large enough to provide sufficient load balancing. This is the main idea behind modified BP. This idea is supported in Figure 2.2 which shows the the shortest and extended paths for a source-destination pair in a random topology. The total number of shortest and extended paths is significantly larger than the number of shortest paths alone. As we can see in the figure, the extended paths allow more node-disjoint routes to be used which is particularly beneficial from a load balancing point of view.

The implementation of Modified BP scheme is straightforward. A node runs an additive metric-based routing protocol (the metric is hop count in our case) and fills in the routing table. Backpressure-based packet forwarding algorithm first selects a destination with the

highest differential queue backlog, and then it selects a next-hop from routing table to forward packets of that destination. We reiterate that Modified BP can work with any additive metric based routing and hop-count based routing is a special case.

2.3 Simulation Results and Analysis

We simulate a time slotted system. The simulation is performed for 15000 slots. We select two source-destination pairs randomly from the network and create flows for them. The traffic is modeled as poisson process. The transmission rate of every link is normalized to 1 packet/slot. In addition to Pure BP, we consider two more routing schemes for comparison purposes. In current systems, most of the routing schemes utilize only a single shortest path which is kept fixed throughout the routing operation until this path gets broken. Thus it is worthwhile to compare the performance of the proposed scheme with single path routing schemes. Assuming time-invarient channel conditions throughout the simulation period, we consider a routing scheme which arbitrarily selects a shortest path from the list of available shortest paths and always sticks to it during the whole simulation period. We refer to this scheme as single path (SP) routing scheme. In all three schemes (Pure BP, Modified BP and SP), we run greedy scheduling for medium access operation. The last scheme is based on single path routing but it uses Aloha-type medium access. We call it *pure* single path routing (Pure SP) since it is the closest version of routing/sheduling schemes seen in current systems today. We compute average queue backlog performance, which is analogous to throughput performance, and end-to-end delay performance of these four schemes for different popular network topologies. *Note that even though the presented results are specific to a given topology, they represent a common trend among various sample topologies we tested in our simulation.*

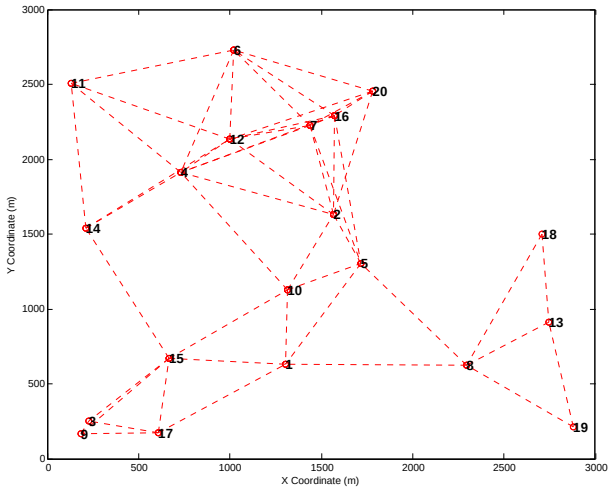

Figure 2.3: Sample uniform random topology

2.3.1 Average Queue Backlog Performance

First, let us consider the 20 node random topology shown in Figure 2.3, where the nodes are randomly distributed in 3000 x 3000 m square plane. Figure 2.4 presents the average queue backlog performances of given schemes in this network. The traffic load on the x-axis is presented in terms of number of packets per second per flow i.e. the average number of packets each source generates in every second. The average queue backlog performance represents the traffic load which a network can support with no queue leading to infinity. At low traffic load, queue lengths in the network are small. But as the traffic load increases, more unserved packets accumulate in the queues and the average queue length rises. After some *critical load*, it becomes impossible to serve any higher load and the average queue length shoots to infinity. This critical load marks the capacity of a network under given scheme. It can be easily observed that Modified BP performs as well as Pure BP because both of them achieve almost the same network capacity. SP pays the penalty of using a fixed single path and its performance is moderate. As expected, Pure SP shows the worst performance and gets saturated under light traffic loads, primarily because it does not use backpressure

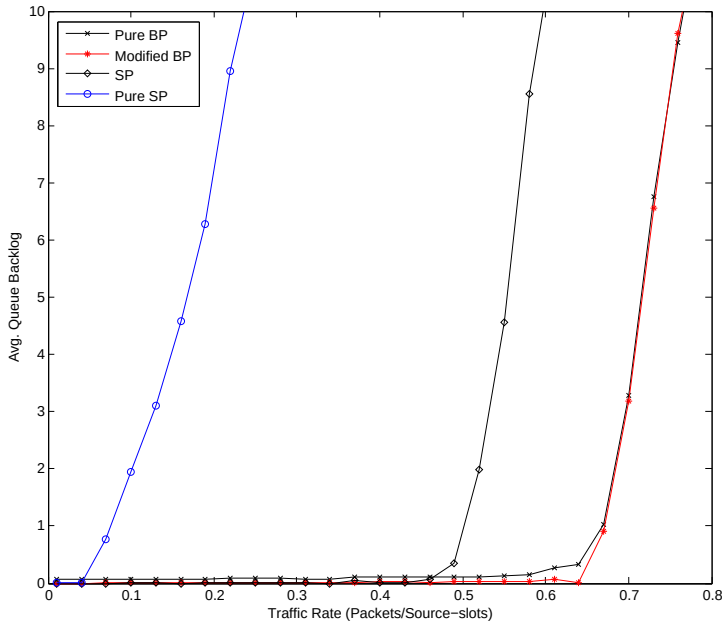

Figure 2.4: Average queue backlog performance-comparison for 20 node random topology

scheduling and multiple routes for load balancing. We reiterate that these results present a common trend among most of the sample topologies we considered in simulation.

Next, we consider the 20 node grid topology shown in Figure 2.5. Note that in a grid topology, extended paths do not exist due to the structure of this topology. Thus Modified BP can use only shortest paths in this case. Figure 2.6 presents average queue backlog performance of these schemes in the given grid topology. The performance of Pure BP is superior to Modified BP, which is primarily due to the absence of extended paths. But still, Modified BP performs significantly better than the SP and Pure SP routing scheme. The behavior of other schemes in the grid topology is consistent with the results for random topologies.

In simulation results, we see that Modified BP performs as well as Pure BP for random topologies. The intuitive reason behind this trend is that if we have sufficient number of shorter paths, using long paths does not provide any significant benefits, even when the shorter paths are congested. Consider an example where we have a sufficient number of shorter routes for a source-destination pair. Let us assume that all of them get congested and source starts forwarding the traffic on longer routes. Assuming the same traffic conditions as before, it can be easily understood that the traffic load which saturated shorter routes

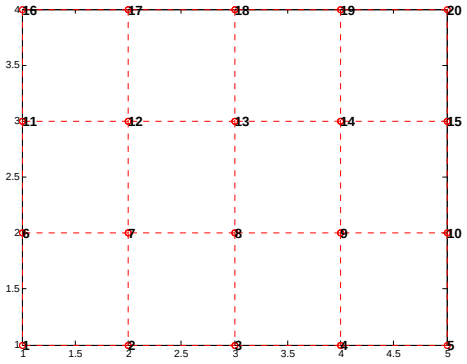

Figure 2.5: Sample grid topology

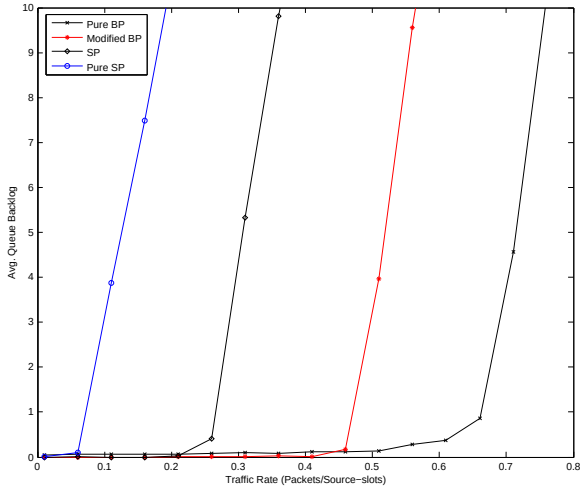

Figure 2.6: Average queue backlog performance-comparison for 20 node grid topology

before, would congest longer routes sooner or later. Thus most of the packets on long routes would remain in the network for a long time. In the mean time, due to faster service time of shorter routes, some shorter routes would become available to serve new packets. Thus long routes provide little overall benefit. But the routes which are slightly longer than shortest paths e.g. the extended paths, can still help to alleviate the congestion because delay on such routes is comparable to the delay on the shortest paths. Thus if such routes are sufficient in number, we don't need to forward traffic on much longer paths like Pure BP case. We have already shown that generally, the total sum of the shortest paths and extended paths is sufficiently large in a random topology. This explains the good performance of proposed scheme in the random topology. This also explains its moderate performance in the grid topology as it could only use the shortest paths which were small in number.

As a side note, when a network is uniformly loaded, using even extended paths does not provide much gain. We illustrate this through the performance plots in Figure 2.7. We simulate a scenario in a random topology where every node has some traffic for every other node in the network. We still use Modified BP but disable all extended paths and forward traffic on the shortest paths only. Let us call it Modified BP with shortest paths for the sake of this discussion and compare it with Pure BP and SP schemes. We see that utilizing the shortest paths only provides the performance as well as Pure BP in this scenario. The same trends are seen as when we simulated a grid topology. These results suggest that when a node uses a longer path, it may create the congestion for other flows using that path as their shortest path already. This may force these flows to switch to other longer paths which might be the shortest path for the original flow. Thus in this 'tit for tat' scenario the original flow gets no gain from using longer paths. (In fact, the situation worsens from the delay point of view.) But Modified BP with shortest paths still achieves load balancing gain due to multiple shortest paths compared to the SP scheme because SP scheme shows poor performance when compared to it.

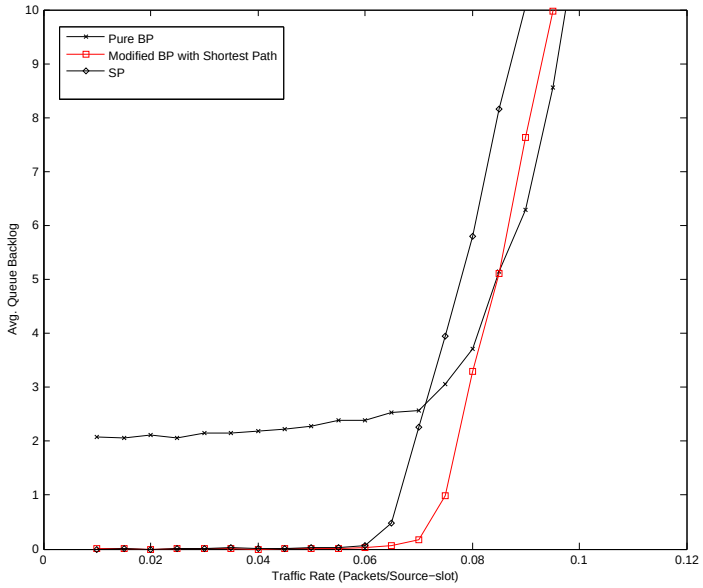

Figure 2.7: Average queue backlog performance-comparison in uniform traffic load scenario

2.3.2 End-to-End Delay Performance

Though the average queue backlog plots give some insight into delay properties of the network, the average end-to-end delay performance is still not clear. To plot end-to-end delay performance, we generate traffic for 15000 slots and run the simulation until all packets in the network have been served. Figure 2.8 presents the end-to-end delay performance of above-mentioned schemes in the random topology. At low traffic loads when end-to-end delay due to path traversal supersedes the queuing delay, Pure BP has *unnecessary* high end-to-end delay because it sends many packets on long routes even when a single shortest path is able to accommodate the traffic load. Other schemes including Modified BP have almost negligible delay in light load conditions because they always use the shortest or slightly longer paths. At higher traffic loads, queueing delay dominates over path traversal delay. Thus, SP and Pure SP have poor delay performance when compared with Pure BP because the queues under these scheme saturate at much less load when compared to Pure BP. But Modified BP manages to perform as well as Pure BP at high traffic loads. The reason can be explained on the basis of average queue length performance of Modified BP. At high traffic loads, the queues in Modified BP scheme have almost the same average length that they do in Pure BP.

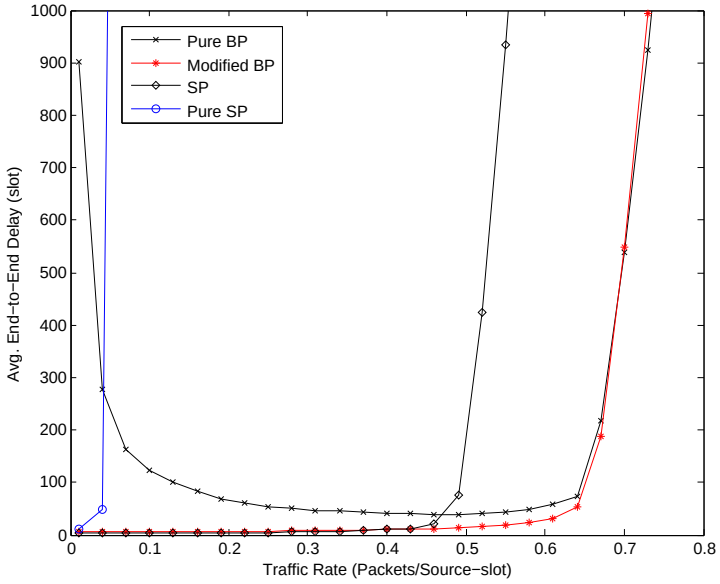

Figure 2.8: Delay performance results for random topology

Thus the queuing delay in both cases remains almost the same and approximately equal to the average end-to-end delay under high traffic load. Thus across all traffic loads, modified BP has superior end-to-end delay performance compared to other schemes. Under light load it outperforms Pure BP, while under high load it outperforms SP and Pure SP and performs close to Pure BP.

In addition to *average* delay performance, it is worth looking into end-to-end delay distribution of packets under Pure BP and Modified BP scheme. These results are presented in Figure 2.9. The distribution for Pure BP has to be truncated as few packets experience an end-to-end delay of more than 2000 slots. Nevertheless, it is clear from the figure that the delay distribution for Pure BP has much longer tail than what is seen in Modified BP case. Most of the packets in Pure BP have an end-to-end delay between 20 and 200 slots, while in Modified BP most of the packets experience an end-to-end delay of no more than 20 slots.

Our final observation is about the relationship between topology control and the throughput obtained by BP algorithm. Generally, dense topologies may have more shorter paths for a source-detsination pair when compared with sparse topologies. Thus the throughput performance in dense topologies might be better than sparse topologies. This gives an insight

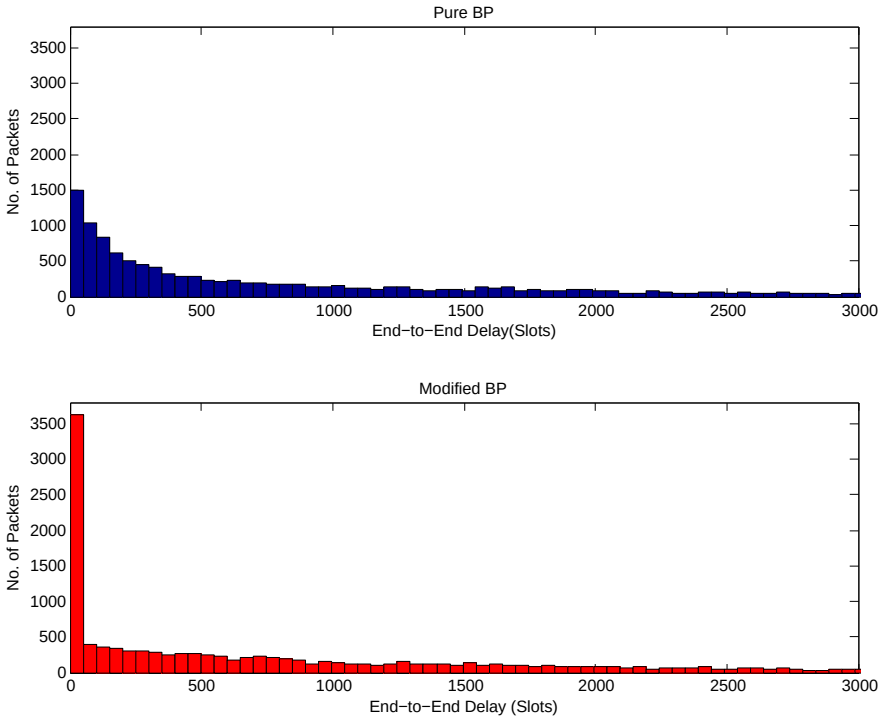

Figure 2.9: End-to-end delay distribution of packets for Pure BP. Traffic rate is 0.5 packets/user - slot

into the topology control problem: one should try to obtain a dense topology if one wants to maximize the throughput by using a scheme similar to backpressure. This is an interesting observation and is a subject to our future study.

2.4 Conclusion

We discussed delay-related issues in throughput-optimal classical backpressure routing. We targeted these issues and proposed a scheme which uses backpressure for packet forwarding while route discovery is taken care by a generic routing protocol. Our approach uses routes from two categories: The shortest paths and the extended paths. Results show that our approach performs close to classical backpressure routing in throughput region while the delay performance. The delay performance of our scheme is significantly superior than classical backpressure scheme. Overall, our proposed scheme performs well enough in most cases of practical interest. We also show that under uniform traffic load scenarios, forwarding packets on longer routes does not give any tangible benefit and utilizing only the shortest paths is best strategy.

There are still some practical issues which remain unaddressed in this chapter. Under light traffic load when utilizing a shortest path is sufficient, our scheme may still choose an extended path to forward the traffic. Though, less delay penalty has to be paid in utilizing the extended paths, using them only when required would be a further enhancement of our scheme. Another issue is the estimation of differential queue backlog with respect to all one hop neighbors. In our simulation, the instantaneous value of differential queue backlog is assumed to be accurately known in every time slot. In practice, due to messaging overheads, it may not be advisable to update the differential queue backlog frequently.

Chapter 3

Backpressure Scheduling in Realistic Interference Model

A scheduling algorithm makes medium access decision for a node. The performance of a scheduling algorithm strongly depends upon the interference model chosen. An interference model dictates the interference relationship between any two nodes in the network. There are three most common interference models used: The hop-based model, the disk-based model and the standard physical model, also referred as the signal to noise ratio (SINR) model. A K-hop-based interference model assumes no interference after K-hops. Thus all the transmitting nodes who are away from a node by more than K-hops, do not interfere the node under this model. The disk-based models are a variation over hop-based models. A disk-based interference model defines an interference range for a node. A transmitter does not interfere with a node if it lies outside the interference range of the transmitter. These models inherently assume that interference ends abruptly at some boundary and does not propagate further. This model underestimates the interference by ignoring the accumulative interference due to far away transmissions. Thus, the actual performance of any general policy (including scheduling) using these interference model may be poorer than its claimed performance [54]. The third interference model, the SINR model captures more realistic

properties of interference than previous two interference models discussed. (Though it ignores other effects such as shadowing and fading.) This model accounts for every possible interferer and assumes a successful transmission only when SINR at the receiver is greater than some threshold.

Past works on scheduling exploit graph-theoretic techniques in scheduling policy design e.g. [8,10,17]. They model the scheduling problem as finding maximum weighted independent set (MWIS) of a *binary conflict graph*. The binary conflict graph represents a binary interference relationship among all links in the network and suggests whether any link pair in the network can be activated simultaneously or not. Abstracting a network as a binary conflict graph is useful for simple protocol design for higher layers. Most of the past scheduling works adopt either a hop-based or a disk-based interference models for conflict graph approach, but these models have their own disadvantages as discussed before. The scheduling works dealing with SINR model adopt linear and non-linear programming approaches [40–42] which are not computationally efficient even for moderate size networks. However, the SINR model is not adopted for binary conflict graph based scheduling. Though graph-level abstraction of a network under SINR model is possible using interference graph approach [42, 45]. In the interference graph, each node has a weighted edge to every other node in the network. The weight represents the amount of power one node receives from the other. Scheduling in an interference graph requires information from all the links while adding or removing a single link in schedule. Besides this drawback, the scheduling complexity, i.e. schedule computation time, in the interference graph is substantially higher than in a binary conflict graph. Unfortunately, establishing binary conflict relationship between link pairs is not straightforward in the SINR model; it is not even clear that it is possible, except trivially. This is one reason why past scheduling policies in the SINR model do not utilize the well-developed class of powerful graph-theoretic techniques that apply to the conflict graph-based interference model. Thus the hop-based or disk-based and the SINR model both have their own benefits from scheduling point of view. In order to gain the advantages of both models, it would be ideal to capture SINR properties in the binary conflict graph. This chapter is

based on the same idea. We propose a synthetic interference model that keeps backpressure scheduling design simple and also accounts for the realistic interference. We show how to construct a binary conflict graph satisfying SINR constraints in the uniform and non-uniform power assignment cases. Scheduling policies relying on the SINR-based conflict graph could be fast, simple, and have provable SINR guarantees. These policies also have potential to be implemented in a distributed manner. This work is also presented in [55].

The rest of this chapter is organized as follows. In section 3.1, we formally describe our network model. Sections 3.2 and 3.3 explain the construction of a conflict graph in power control and uniform power settings respectively. Section 3.4 proves the validity of the obtained schedule. The simulation model and results are discussed in section 3.5. Finally, we conclude the chapter. Note that we deal with two different network graphs in the chapter: the communication graph and conflict graph. To avoid any confusion, we will refer to nodes and links for the communication graph while referring to vertices and edges for the conflict graph. A vertex in the conflict graph represents a link in the communication graph. An edge between two vertices implies that the corresponding links conflict with each other.

3.1 Network Model

Let N be the set of n transceivers or nodes distributed arbitrarily in the plane. Let d_{ab} be the Euclidean distance between nodes a and b. Suppose, node a is transmitting to the node b with transmit power P_a. Then, the received signal power at node b due to node a is $P_{ab} = P_a/d_{ab}^{\alpha}$ according to the exponential path loss model where α is the path loss exponent, assumed to be greater than 2. Let us define a link $e = (a, b)$ if and only if $P_{ab}/N_0 \geq \beta$ where N_0 is the ambient noise level and β is the SINR threshold. For rest of the chapter, the transmitter and receiver of a link e_i will be denoted by a_i and b_i respectively. Assuming that initially every node transmits with its maximum power, define E as the set of all links present in the network. Define $G = (N, E)$, a directed communication graph, and assume that G is connected. Consider $S = \{a_1, a_2, \ldots a_k\}$ as a set of transmitters active at the time

when node a is transmitting to node b. The SINR at b can be given as

$$\text{SINR}_{ab} = \frac{P_{ab}}{N_0 + \sum_{a_i \in S, a_i \neq a} P_{a_i b}}$$

The transmission from a is received *correctly* at b if $\text{SINR}_{ab} \geq \beta$. Given the uncertain nature of wireless link, acknowledgement (ACK) from a receiver to transmitter is usually required to provide reliable link layer operation. We assume that a packet sent by node a is *successfully* received by node b if and only if the packet is correctly received by b and the ACK sent by node b is correctly received by node a. The role of this assumption will be clear later in section 3.2. We also assume that a node cannot transmit and receive simultaneously, the so-called self-interference constraint. Neither can a node receive from or transmit to more than one node at a time.

3.2 Constructing Conflict Graph with Power Control

As mentioned before, we assume bi-directional links and hence consider the interference due to ACK transmission by receivers also. Thus, for a transmission on $q = (x, y)$ that is concurrent with the transmission on $p = (a, b)$, we account for the interference both from node x's data packet and from node y's ACK. Since only one of x and y transmits at a time but their data and ACK packets could both overlap with either the data packet or the ACK along $p = (a, b)$, we have to choose the maximum of the interferences from x and y when accounting for the effective interferences at a and b. Note that which of the two (x or y) contributes the maximum interference could be different at a and b. Thus to determine the conflict between p and q, distances between four pairs must be considered: $(a, x), (a, y), (b, x)$, and (b, y). Let us define the term *link distance*, somewhat strangely, as the minimum distance among all four pairs.

Definition 1. *Given two links $p = (a, b)$ and $q = (x, y)$, link distance D_{pq} is defined as minimum of all node pairs excluding the node pairs which constitute the links themselves. Hence, $D_{pq} = \min\{d_{ax}, d_{ay}, d_{bx}, d_{by}\}$*

First, we describe our idea of forming a conflict graph in the SINR model. We say that two links e_i and e_j conflict with each other if $D_{e_ie_j}$ is less than R_{ij}. As shown later in the section 3.2.2, R_{ij} is a function of link lengths denoted by l_i and l_j. We will show if $D_{e_ie_j}$ is at least R_{ij}, the transmission on both links e_i, e_j would be successful. After determining the conflict relationship between link pairs, we can construct a binary conflict graph for G.

3.2.1 Set Partitioning

We partition the set E into subsets based on the link lengths. This partition carries importance in determining the power levels as shown later in the power control part. Let us define $\delta = d_{\max}/d_{\min}$ where $d_{\max}$ and $d_{\min}$ are the maximum and minimum link lengths in the network. The *SetPartition* algorithm, algorithm 1, partitions E into at most $\lceil \log \delta \rceil + 1$ disjoint sets. A set B_m contains every link $e_i \in E$ such that $2^m \leq l_i \leq 2^{m+1}$ for $m = \{0, 1, \ldots, \lceil \log \delta \rceil\}$. After initial partitioning of the links into sets, all empty link sets are removed and remaining sets are arranged in the descending order of the link lengths (line 3). Let τ_i denote the *length class* of link e_i. That is if $e_i \in B_m$ then $\tau_i = m$ (line 7).

Algorithm 1: SetPartition

Input: $G = (N, E)$

Output: A partition B of E.

1 Partition the set E into sets $B = \{B_0, B_1, \ldots, B_{\lceil log\delta \rceil}\}$ such that $B_m = \{e_i : e_i \in E \text{ and } 2^m \leq l_i < 2^{m+1}\}$

2 If B_m is empty, remove B_m. Rename set B such that B_m is the m^{th} non-empty set in *decreasing* order of the link lengths

`// set index of link` e_i `as set index` m

3 for each $e_i \in B_m$, assign $\tau_i = m$

3.2.2 Conflict Graph Formation and Power Control

The *ConstructConflictGraph* algorithm, algorithm 2, picks one link pair (e_i, e_j) at a time. From section 3.1, the self-interference property implies that adjacent links cannot be active simultaneously. Thus, if e_i and e_j are adjacent (line 4), they conflict with each other. If they are not adjacent, we need to check further if they conflict. If e_i and e_j are from the same length class i.e. $\tau_i = \tau_j$, R_{ij} is given by (line 9)

$$R_{ij} = \mu \min\{l_i, l_j\} \tag{3.1}$$

The value of μ is given in *ConstructConflictGraph* (line 1). If τ_i and τ_j are different, then R_{ij} is defined as (line 11)

$$R_{ij} = \max\{l_i, l_j\}(3n\beta)^{(|\tau_i - \tau_j|+1)/\alpha} \tag{3.2}$$

Finally, we calculate D_{ij} and if it is less than R_{ij}, we say that e_i and e_j conflict with each other. In case of conflict, we join the corresponding vertices of a link pair by an edge in the conflict graph K. After enumerating all link pairs, we get our final conflict graph.

Power control is the last stage. The power level assigned to both nodes a_i and b_i of a scheduled link e_i is $P_{e_i} = P_{a_i} = P_{b_i} = 4N_0(3n\beta)^{\tau_i} l_i^{\alpha}$. This would give us power allocation vector for the given schedule. Though, power control appears separate from the conflict graph formation, there is a close relationship between the given value of R_{ij} in *ConstructConflictGraph* and assigned power level for any link. As the power allocation strategy suggests, the links of smaller lengths might be allocated higher power as compared to the links of bigger lengths. This enables the smaller links to cope with potentially higher interference from bigger links. A similar power control scheme is used in [56]. However, their algorithm can only schedule links from the same length class.

Algorithm 2: ConstructConflictGraph

Input: $G = (N, E), B$

Output: A conflict graph K

$\mu = 2^{\frac{2}{\alpha}+2}\left(\frac{\beta(\alpha-1)}{\alpha-2}\right)^{1/\alpha}$

for $i = 1$ *to* $|E|$ **do**

  for $j = 1$ *to* $i - 1$ **do**

    `// Self-interference constraints`

    if $a_i = a_j$ *or* $b_i = b_j$ *or* $a_i = b_j$ *or* $b_i = a_j$ **then**

      e_i and e_j conflicts

    else

      `// Determining if links` i `&` j `conflicts`

      if $\tau_i = \tau_j$ **then**

        $R_{ij} = \mu \min\{l_i, l_j\}$

      else

        $R_{ij} = \max\{l_i, l_j\}(3n\beta)^{(|\tau_i - \tau_j|+1)/\alpha}$

      $D_{ij} = \min\{d_{a_i a_j}, d_{a_i b_j}, d_{b_i a_j}, d_{b_i b_j}\}$

      if $D_{ij} < R_{ij}$ **then**

        e_i and e_j conflicts

  end

end

3.3 Constructing a Conflict Graph with Uniform Power

Section 3.2 considers power control to construct a conflict graph. In this section, we discuss the binary conflict graph formation in uniform power settings. The basic approach is similar to what is used in the above section i.e. if two links are sufficiently far apart, they can be scheduled simultaneously.

The idea is to follow the same *SetPartition* procedure given in section 3.2.1. Then, applying the constraints given in *ConstructConflictGraph* (lines 4 and 7), the links from different length classes can be scheduled. Precisely, any two links i, j can be scheduled if D_{ij} is greater than $\mu \max\{l_i, l_j\}$. Otherwise, they conflict with each other. The algorithm is presented in procedure *UniPowConflictGraph*. A similar scheme was recently presented in [57] but that scheme is used to prove the NP-hard nature of scheduling in the SINR model and not used for conflict graph formation.

Algorithm 3: UniPowConflictGraph()

Input: $G = (V, E), B$

Output: A conflict graph K

1 $\mu = 2^{\frac{2}{\alpha}+2} \sqrt[\alpha]{\frac{\beta(\alpha-1)}{\alpha-2}}$
2 **for** $i = 1$ *to* $|E|$ **do**
3 **for** $j = 1$ *to* $i - 1$ **do**
  `// Self-interference and length-class constraints`
4 **if** $a_i = a_j$ *or* $b_i = b_j$ *or* $a_i = b_j$ *or* $b_i = a_j$ *or* $\tau_{\mathbf{i}} \neq \tau_{\mathbf{j}}$ **then**
5 e_i and e_j conflicts
6 **else**
7 $D_{ij} = \min\{d_{a_i a_j}, d_{a_i b_j}, d_{b_i a_j}, d_{b_i b_j}\}$
8 **if** $D_{ij} < \mu \min\{l_i, l_j\}$ **then**
9 e_i and e_j conflicts
10 **end**
11 **end**

3.4 Analysis of Policies

3.4.1 Power Control Approach

We prove that all transmissions on a set of links will be successful provided that the chosen set of links is an independent set with respect to the conflict graph, i.e. there are no pairwise conflicts within the set. The proofs use similar methods to those in [43]. The outline of the proof is as follows: In Lemma 1 and Lemma 2, we bound the interference generated by simultaneous active links of different length classes. In Lemma 3, we bound the interference due to simultaneous active links of the same length class. Finally, using these lemmas in theorem 1, we get the upper bound on the total interference accrued at any node of a link and prove that that the resulting SINR is always less than β. Before presenting proofs, let I_s^i be the interference experienced at link e_s due to link e_i. We define I_s^i as follows:

$$\begin{aligned} I_s^i &\leq \max\{P_{a_i a_s}, P_{b_i a_s}, P_{a_i b_s}, P_{b_i b_s}\} \\ &= P_{e_i}/D_{e_i e_s}^{\alpha} \end{aligned}$$

Lemma 1. *Consider a link e_s scheduled in a time slot t. Let e_i be a simultaneously scheduled link with $\tau_i < \tau_s$, then*

$$I_s^i \leq \nu(3n\beta)^{\tau_s - 1}$$

Proof. For $\tau_i < \tau_s$, $D_{e_i e_s} \geq (3n\beta)^{(\tau_s - \tau_i + 1)/\alpha} l_i$. Thus

$$\begin{aligned} I_s^i &\leq \frac{\nu(3n\beta)^{\tau_i} l_i^{\alpha}}{(3n\beta)^{(\tau_s - \tau_i + 1)} l_i^{\alpha}} \\ &\leq \frac{\nu(3n\beta)^{\tau_i} l_i^{\alpha}}{l_i^{\alpha}} \\ &= \nu(3n\beta)^{\tau_i} \\ &\leq \nu(3n\beta)^{\tau_s - 1} \end{aligned}$$

□

Lemma 2. *Consider a link e_s scheduled in a time slot t. Let e_i be any simultaneously scheduled link with $\tau_i > \tau_s$, then*

$$I_s^i \leq \nu(3n\beta)^{\tau_s - 1}$$

Proof. Given $\tau_i > \tau_s$, $D_{e_i e_s} \geq (3n\beta)^{(\tau_i - \tau_s + 1)/\alpha} l_i$. Thus

$$\begin{aligned} I_s^i &\leq \frac{\nu(3n\beta)^{\tau_i} l_i^\alpha}{(3n\beta)^{(\tau_i - \tau_s + 1)/\alpha} l_i^\alpha} \\ &\leq \nu(3n\beta)^{\tau_s - 1} \end{aligned}$$

□

Lemma 3. *Consider a link $e_s = (a_s, b_s)$ scheduled in a time slot t. Let I'_s be total interference due to all simultaneously active links of the same length class, then*

$$I'_s \leq \frac{48\nu(3n\beta)^{\tau_s} 2^\alpha}{\lambda^{\alpha-1}\mu^\alpha}$$

Proof. Two links l_i, l_s of same length class would be scheduled only if $D_{e_i e_s} \geq \mu \min\{l_i, l_s\}$. Since l_i and l_s belong to same length class, $D_{e_i e_s} \geq \mu l_s/2$ irrespective of which is bigger, l_i or l_s. Assume a set S' containing all activated links of same length class. So, all transmitting nodes in S' must be far from b_s at least by $\mu l_s/2$. Thus, the disks of radius $\mu l_s/4$ centered at all transmitting nodes do not overlap. Next, to calculate the total interference at b_s due to set S', we partition the plane into rings R_λ centered at b_s and of width μl_s for $\lambda = 1, 2, \ldots$. R_λ contains all transmitters for which $\lambda\mu l_s \leq d_{b_s a_i} < (\lambda + 1)\mu l_s$. The area of a ring R_λ can be calculated as

$$\begin{aligned} A(R_\lambda) &= \pi[((\lambda + 1)\mu l_s)^2 - (\lambda \mu l_s)^2)] \\ &= \pi\mu^2(2\lambda + 1) l_s^2 \\ &\leq 3\pi\lambda\mu^2 l_s^2 \end{aligned}$$

Calculating total number of transmitting nodes in R_λ

$$\frac{3\pi\lambda\mu^2 l_s^2}{\pi\mu^2 l_s^2/16} \leq 48\lambda \tag{3.3}$$

The maximum possible transmit power of any node in S' can be $\nu(3n\beta)^{\tau_s}(2l_s)^{\alpha}$. Thus, the cumulative interference at b_s due to transmitters in R_λ

$$\begin{aligned} I_\lambda &\leq 48\lambda \frac{\nu(3n\beta)^{\tau_s}(2l_s)^{\alpha}}{(\lambda\mu l_s)^{\alpha}} \\ &\leq \frac{48\nu(3n\beta)^{\tau_s}(2)^{\alpha}}{\lambda^{\alpha-1}\mu^{\alpha}} \end{aligned}$$

Summing up the total interference due to transmitters in R_λ for every λ

$$\begin{aligned} \sum_{\lambda=1}^{\infty} I_\lambda &= \frac{48\nu(3n\beta)^{\tau_s}(2)^{\alpha}}{\mu^{\alpha}} \sum_{\lambda=1}^{\infty} \frac{1}{\lambda^{\alpha-1}} \\ &< \nu n^{\tau_s}(3\beta)^{\tau_s-1} \end{aligned} \tag{3.4}$$

The last inequality follows from the definition of μ and the result $\sum_{\lambda=1}^{\infty} 1/\lambda^{\alpha-1} = (\alpha - 1)/(\alpha - 2)$ for $\alpha > 2$. □

Next we prove in the following theorem that all links in a schedule satisfies SINR constraint.

Theorem 1. *If any set of links forming an independent set in the conflict graph is activated simultaneously, then each link in the set will satisfy its SINR requirement.*

Proof. For any receiver on a link, at most $n - 2$ nodes can interfere. Thus, from Lemma 1 and Lemma 2, maximum total interference from the links of different length classes can be given as

$$I_{diff} = (n-2)\nu(3n\beta)^{\tau_s-1} \leq n\nu(3n\beta)^{\tau_s-1}$$

From Lemma 3, the total interference caused by all active links from the same link class as link s is bounded by $\nu n^{\tau_s}(3\beta)^{\tau_s-1}$. By summing up the these interference values, we can get the bound on total interference accrued at the receiver

$$\sum_i I_s^i < \frac{2\nu}{3}\beta^{(\tau_s-1)}(3n)^{\tau_s} \tag{3.5}$$

From (3.5), the SINR experienced at receiver b_s of link e_s will be at least

$$\begin{aligned}\text{SINR}_{b_s} &> \frac{4N_0(3n\beta)^{\tau_s}}{N_0 + \frac{2\nu}{3}\beta^{(\tau_s-1)}(3n)^{\tau_s}} \\ &= \frac{4(3n\beta)^{\tau_s}}{1 + \frac{8}{3}\beta^{(\tau_s-1)}(3n)^{\tau_s}} \\ &\geq \frac{4(3n\beta)^{\tau_s}}{\frac{11}{3}\beta^{(\tau_s-1)}(3n)^{\tau_s}} > \beta\end{aligned}$$

The second inequality is obtained by putting $\nu = 4N_0$ and third inequality follows from the fact that $n, \beta > 1$. This holds for every link in the independent set. Thus, if a set of links from an independent set in the conflict graph is activated simultaneously, then the transmission over each link in the set will always be successful. This proves our theorem. □

3.4.2 Uniform Power Approach

Next, we prove that under uniform power assignment, the SINR constraint for any link in the obtained schedule will always be satisfied.

Theorem 2. *If any set of links forming an independent set in the conflict graph is activated simultaneously in uniform power case, then each link in the set will satisfy its SINR requirement.*

Proof. We can construct the proof using intermediate results obtained in Lemma 3. Suppose, we activate a link from length class m i.e. for any link e in class m, $2^m \leq l_e < 2^{m+1}$. Let the transmit power of any node is P. Consider the worst case scenario when all interferers are at the least possible distance from the receiver b_s i.e. 2^m. Using similar arguments as in Lemma 3, maximum possible interference due to all interfering nodes can be bounded as

$$I < \frac{3P}{2^{2\alpha} l_s^{\alpha} \beta} \tag{3.6}$$

Next, we calculate the worst-case SINR for link l_s. The worst-case SINR is given as

$$\text{SINR}_{b_s} = \frac{P/l_s^{\alpha}}{N_0 + 3P/2^{2\alpha} l_s^{\alpha} \beta} \tag{3.7}$$

If we assume l_s to be less than or equal to $[P(1-3/2^{2\alpha})/\beta N_o]^{1/\alpha}$, SINR is always greater than β. This should be a fair assumption as it suggests that under worst-case interference, a link has to increase its power by $(1-3/2^{2\alpha})^{1/\alpha}$, a small value for any value of $\alpha \geq 2$. Finally, as explained above, the SINR at both nodes of every link will always be higher than β. Thus, schedule over the obtained conflict graph will always be valid. □

3.5 Simulation Model and Results

Nodes are randomly distributed in 3000 x 3000 m plane. The transmitter power for uniform power level is 10 dBm and N_0 is -90 dBm. The values of path loss exponent α and threshold β are taken as 3 and 10 dB respectively. The simulation is performed for 50000 slots. In every slot, packets are generated at every node as a poisson random process and destination is assigned randomly among all nodes uniformly. The rate of every link is normalized to 1 packet/slot. Backpressure routing is used.

3.5.1 Baseline Interference Models

In the literature, there are two popular graph based interference models: the two-hop interference model and the physical interference graph. Two-hop interference model is commonly used for IEEE 802.11 type of wireless networks because it is a good model of the functioning of the RTS/CTS mechanism. Under two-hop interference model, each link conflicts with all of its two-hop neighboring links. Further, greedy scheduling can be applied over this conflict graph. On the other hand, while applying greedy scheduling over interference graph, a link with highest backlog is selected first. It is put into the schedule by removing its transmitter and receiver node from the interference graph and adding incurred interference to all other nodes. Next, a new link is scheduled from the modified interference graph, only if 1) it does not cause the SINR of prescheduled links to go below β and b) SINR at this link is not less than β. This continues until no vertices remain in the physical interference graph.

We compare the performances of our proposed approaches with the performances of scheduling in two-hop model (*Ideal Two Hop*), and physical interference graph (*Interference Graph*). We refer to our power control based scheduling and uniform power based scheduling as *Power Control* and *Uniform Power* respectively. Though comparison between power control and uniform power approach is not fair, the performance of power control approach gives insight into the gain one can achieve after applying power control. The performance of greedy scheduling in physical interference graph might be seen as optimal performance for *Uniform Power* case as it is empirically seen that centralized greedy scheduling performs as well as optimum scheduling. We also compute the actual performance of two-hop interference model and compare it with others for illustration purpose. This is done by removing those links from the obtained schedule whose SINR is less than β. We simulate the performances of policies for various arbitrary topologies and present our result for one sample topology as the trends in the results for all taken topologies are similar.

3.5.2 Simulation Results

Fig 3.1 shows the sample topology for which we present our performance results. The performance plot of above discussed policies is given in fig 3.2. The y-axis represents average queue length of node in the network. As we increase the traffic rate for every user simultaneously, the average queue length also increases. But, at some traffic rate, the average queue length jumps to infinity, instantaneously. This traffic rate will mark the network capacity. Since we do our simulation for the limited number of slots, we have to use some threshold for average queue length rather than infinite average queue length as our queue length criteria. In our simulation, we set the threshold equal to 50. Note that we explore one dimension of capacity region only i.e. all nodes have same traffic arrival rates. As expected, the performance of *Ideal Two Hop* is superior to all other policies. Though, the real performance of two hop model (*Real Two Hop*) is worst than any other policy and is quite close to zero. This shows that maximizing the number of scheduled links in two-hop model yields poor performance.

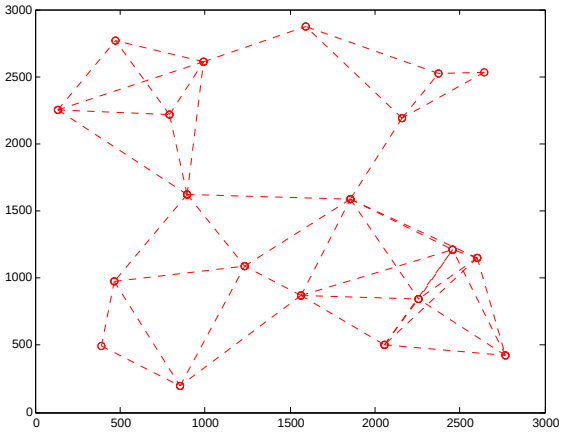

Figure 3.1: Sample network topology

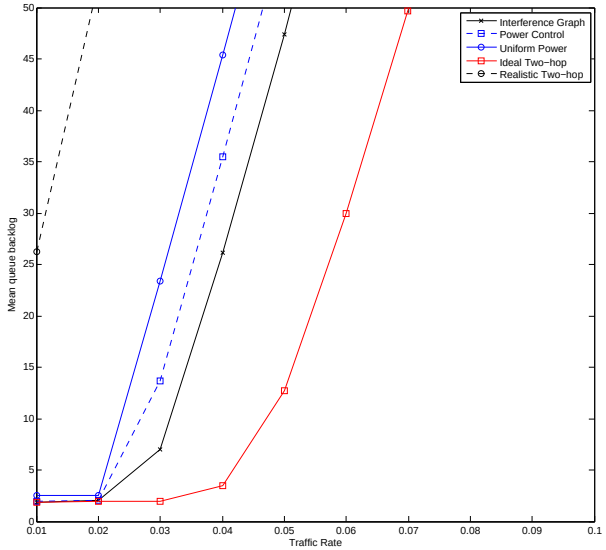

Figure 3.2: Network capacity performance of policies

Of course, the performance of *Real Two Hop* depends heavily on the selection of path loss exponent value. Though, the performance of *Uniform Power* is not close to *Interference Graph*, it achieves a performance ratio of around 0.7 with respect to *Interference Graph* as observed in various topologies. The performance of *Power Control* is close to or slightly better than *Interference Graph* in many cases. Though it is unclear how much better an optimal algorithm for power control case can perform. Note that our main motivation is to present efficient graph based scheduling policies guarantying SINR constraints and there is a good scope to refine these policies to achieve higher throughput further.

Another advantage of SINR based binary conflict graph is proving performance guarantees of policies. In SINR model, it is hard to provide any kind of performance guarantee for any suggested policy. It can be shown that greedy scheduling can perform at least $1/\Delta_{max}$ of the optimal policy for any arbitrary topology where Δ_{max} is the maximum vertex degree of a given conflict graph. These results have been reported in many past scheduling policy based works e.g. [58].

3.6 Conclusion

In this chapter, we presented techniques to construct a SINR-based binary conflict graph using power control and uniform power assignment. The scheduling policy applied over the constructed conflict graphs guarantees that SINR constraints will be met. This work combines the advantages of conflict graph-based scheduling policies with SINR guarantees. The performance of a scheduling policy over proposed conflict graphs is evaluated and compared with other approaches.

The most important issue not discussed in this chapter is the implementation of distributed scheduling policies based on similar ideas. We will discuss the same in the next chapter.

Chapter 4

Distributed Design of Backpressure and Routing Algorithm

In previous chapters, we discussed new designs of backpressure scheduling and routing mechanisms. Chapter 2 proposed some modifications to backpressure routing, while Chapter 3 presented a novel interference model that can be used for backpressure scheduling. The proposed designs are centralized — some omniscient controller takes all the scheduling and routing decisions. These designs cannot be implemented in ad hoc networks in which every node makes decisions by itself with limited knowledge. Thus it is non-trivial to develop distributed algorithms from our centralized designs. We present a distributed design for a backpressure policy for both graph based and SINR-based interference models in this chapter. We simulate these designs and analyze their performance.

4.1 Challenges for Distributed Backpressure Design

Due to the fact that nodes do not have direct knowledge of the queue lengths at their neighbors, scheduling is the major challenge in the design of a distributed backpressure scheme.

Thus a node with light traffic load may block a congested neighboring from transmitting, violating the basic spirit of backpressure. This is against the basic spirit of backpressure. The centralized greedy scheduling selects a node with the highest differential queue backlog first and thus gives higher priority to a congested node for transmission. To implement the similar scheme in distributed and random access based fashion, a node with higher differential queue backlog should be allowed to access the medium with higher probability.

4.2 Distributed Backpressure Algorithm for One-hop Interference Model

We adopt a slotted-Aloha system. First we present distributed backpressure algorithm for one-hop interference model and later extend it to the SINR model. In the one-hop interference model, first introduced in Chapter 2, all disjoint links can be scheduled simultaneously. To exchange queue backlog information with one hop neighbors, we adopt a technique similar to the technique used in [39]. When a packet is transmitted to any destination, the transmitting node piggybacks its queue length for that destination onto the packet. All one-hop neighbors overhear this transmission and update their queue length values for the node-destination pair accordingly.

4.2.1 Neighbor and Destination Selection

We assume an additive metric based generic routing scheme. As in Chapter 2, we use the shortest and the extended paths for packet forwarding. The extended paths are longer than the shortest paths and should be used only when the shortest paths cannot accommodate the current traffic load. Thus, we first use the shortest paths and then, if the current traffic cannot be supported by the shortest paths, we also use the extended paths. For each destination, the node checks the queue length. If the queue length is below a threshold,

QThreshold, the node computes the queue backlog for each neighbors that can reach the destination via a shortest path. If the queue length is above QThreshold, then the node computes the queue backlog for each neighbor that can reach the destination through a shortest or extended path. For all destination-neighbor pairs for which the queue backlog was computed, the node selects the destination-neighbor pair with the highest differential queue backlog. The pseudo-code is presented in Algorithm 4: *DistributedBP*.

4.2.2 Backpressure Based Random Access Scheduling

Once node a selects an optimal neighbor-destination pair (b^*, c^*) and Q_a^* is known, it computes its transmission probability P_a as follows:

$$P_a = 1 - e^{-Q_a^*} \tag{4.1}$$

In practice, P_a can be adapted by changing the contention window size, AIFS values in IEEE 802.11 radios. Though we use the continuous values of P_a in our simulation, the discrete and quantified values can be used for practical systems.

4.2.3 Simulation Model and Results

We simulate our algorithm for Slotted Aloha type system i.e. a time slotted system with random access based MAC. The simulation is performed for 15000 slots. We consider a 40-node random topology where nodes are distributed uniformly in 5000 x 5000 m plane. We select two source-destination pairs randomly from the network and create flows for them. The traffic is modeled as poisson process. The transmission rate of every link is normalized to 1 packet/slot. We have other parameters for our distributed design such as QThreshold. The value of QThreshold is chosen to be 10. It was found that increasing the value of QThreshold to 10 or even 20 did not make any significant impact on the performance. One should choose the high value of QThreshold so that a node should not start using extended

Algorithm 4: DistributedBP()

Input: RoutingTable, ActiveRoutingTable, QThreshold

Output: Tranmission probability for the node, say node a

if $Q^c.Len < QThreshold$ **then**

`// Find the list of neighbors` N_a`, lying on the shortest paths for all destinations`

N_a = RoutingTable.GetShortestPathNeighbors()

else

`// Find the list of neighbors` N_a`, lying on the shortest paths or extended paths for all destinations`

N_a = RoutingTable.GetExtendedPathNeighbors()

end

$(c_{selected}, b_{selected}, DQ_a^*)$ = GetPacketSchedule(N_a, ActiveRoutingTable)

`// Find the optimal neighbor-destination pair` (b^*, c^*) `with maximum differential backlog`

$b^* = \underset{b \in N_a}{\arg\max}\{Q_a^c - Q_b^c\}, \quad \forall c \in V \setminus \{a\}$

$c^* = \underset{c \in V \setminus \{a\}}{\arg\max}\{Q_a^c - Q_{b^*}^c\}$

$DQ_a^* = Q_a^{c^*} - Q_{b^*}^{c^*}$

`// Calculating the probability of transmission`

$P_a = 1 - e^{(-DQ_a^*)}$

paths so early. The higher value of QThreshold will also discourage frequent route switching. The link rate is kept as 1 packet/slot. We simulate two other algorithms to compare with our distributed Backpressure algorithms: Traditional shortest path routing (Pure SP) and centralized backpressure algorithm, both presented in chapter 2 (Modified BP). In Pure SP, all nodes access the medium with same transmission probability and use only one shortest path throughout the whole period of simulation. We compute the average queue backlog and the end-to-end delay performance. The results presented are typical of what we saw over a large number of topologies.

4.2.4 Performance Comparison

Figure 4.1 shows a sample 40-node topology. Figure 4.2 compares the average queue backlog performance of aforementioned backpressure algorithms for the same topology. Not surprisingly, Traditional SP performs quite poorly. But our distributed algorithm performs well and achieves nearly 80% performance of Centralized BP. The loss in performance is expected due to the possibility of random access collisions in Distributed BP; nevertheless, these results are encouraging Next we compare the Distributed BP with Centralized SP algorithm in Figure 4.3. Centralized SP enjoys backpressure based greedy scheduling algorithm but forwards the packet on the shortest paths only and never uses extended paths. In most cases, Distributed BP outperforms Centralized SP. This clearly shows the benefits of forwarding the traffic on extended paths.

Figure 4.4 presents the delay performance-comparison of Distributed BP, Centralized BP and Pure BP. We do not compare them with Traditional SP because it always perform quite poorly. As expected, Distributed BP outperforms Pure BP at light traffic load because it uses shorter routes only as opposed to Pure BP. Of course, Distributed BP cannot support as much traffic as Pure BP and saturates sooner under high traffic conditions. In Figure 4.4, it may appear that Distributed BP and Centralized BP have same end-to-end delay performance. However, Distributed BP is slightly worse than Centralized BP algorithm.

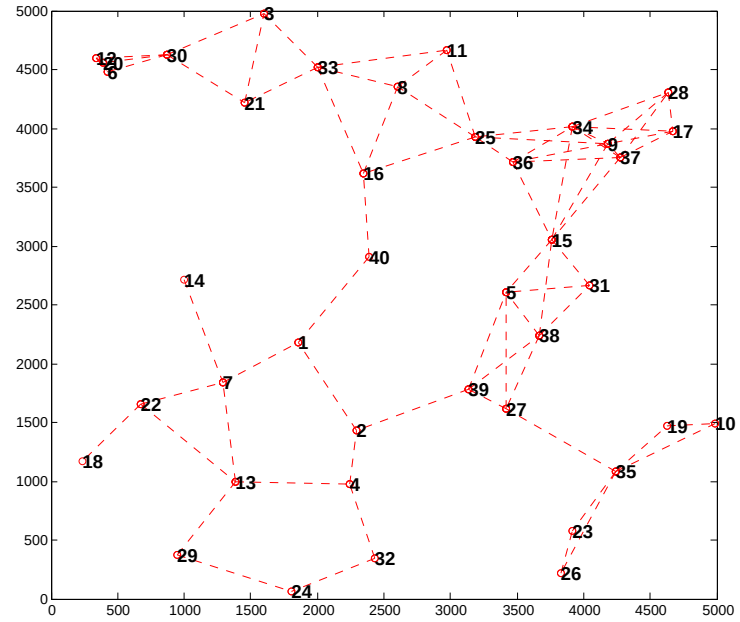

Figure 4.1: A sample uniform random network topology

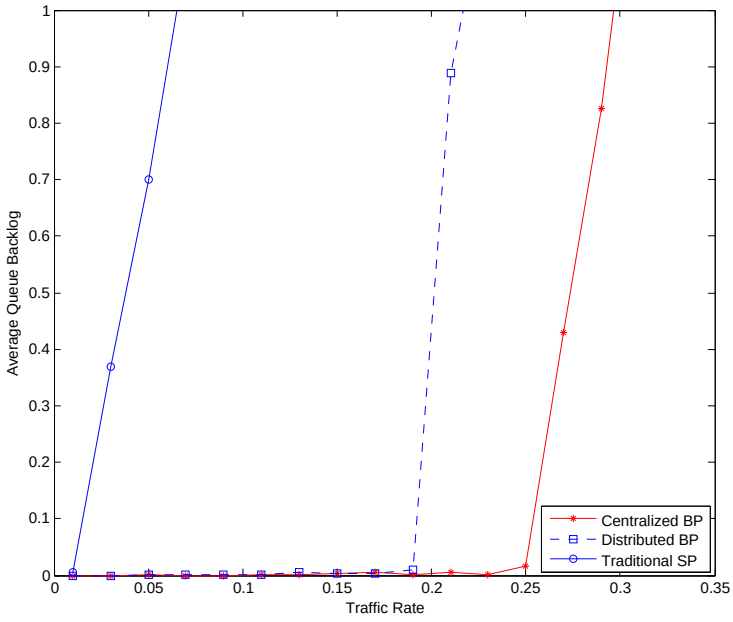

Figure 4.2: Performance-comparison of backpressure algorithms for sample random topology

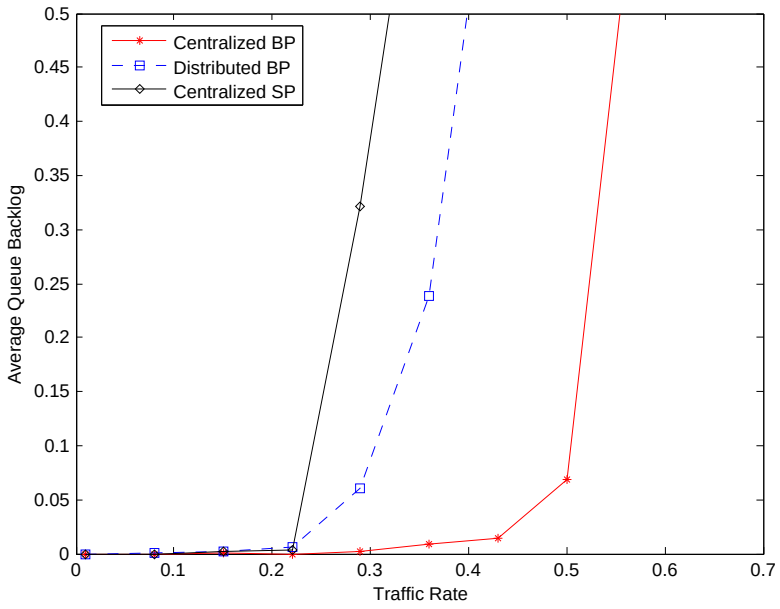

Figure 4.3: Comparison of distributed backpressure algorithm with centralized algorithms

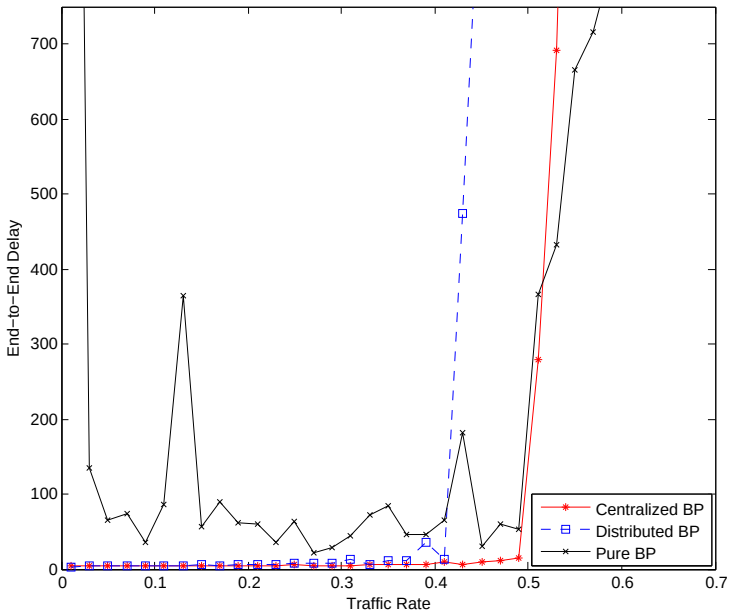

Figure 4.4: End-to-end delay performance-comparison of backpressure algorithms

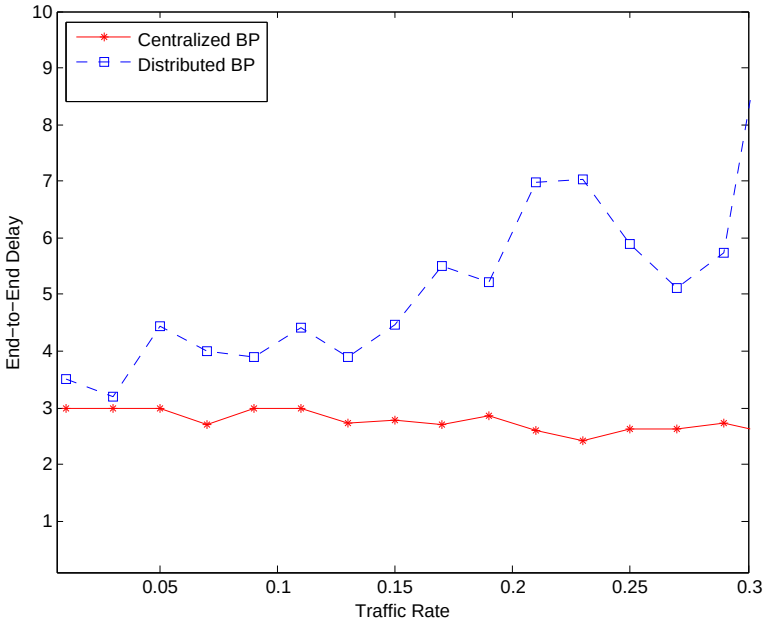

Figure 4.5: End-to-end delay performance-comparison of Distributed BP and Centralized BP

A closer comparison between Distributed BP and Centralized BP is provided in Figure 4.5. The end-to-end delay in Distributed BP is always higher than Centralized BP. This is mainly contributed by higher queuing delay in Distributed BP. The reason being that Distributed BP does not change its route frequently. Thus when some packets stuck into a bad route, they have to wait for longer route time as compared to Centralized BP.

4.3 Distributed Backpressure Algorithm for SINR Model

In the previous section, we presented a design for one-hop interference model. However, as discussed in chapter 3, the SINR interference model is much more realistic than one-hop or even two-hop interference models. We extend our distributed backpressure algorithm from previous section to the SINR model. We assume a Slotted Aloha system and that each node knows its distance from each of its neighbors. An alternative assumption would be that

every node knows the received power level of its transmitted signal at every neighboring receiver. A node can obtain this knowledge by getting feedback from its neighbors. The core part of algorithm is same as before. However, we add extra functionality for SINR measurements into it. This allows a node to determine whether its transmission would be successfully received or not. The basic design of algorithm is as follows:

1. Each node a computes the maximum differential queue backlog and selects the optimal neighbor-destination pair using the procedure *DistributedBP* in section 4.2. It also computes its transmission probability P_a.

2. Then each transmitter sends a control signal to its receiver with probability P_a.

3. Every receiver measures the power level during this period and broadcasts this information after control signal period. The control signal period should be sufficiently long so that power fluctuations can be averaged out. But it should not be too long either, otherwise it would have adverse impact on the throughput.

4. Every transmitter listens back to its receiver and gets the total power sensed at the receiver. If a transmitter doesn't hear a broadcast message for a sufficiently long period, it assumes the receiver acting as a transmitter of some other link and does not do anything further. Otherwise, the transmitter computes knowing interference experienced at the receiver. The interference is calculated by by subtracting the known received power from the total measured power. It further calculates the SINR at the receiver. If SINR is greater than β, it transmits a packet to the receiver. Please note the packet is from the queue of selected destination. If SINR fails to be larger than β, it does not do anything further and wait for next slot.

The control signaling for this algorithm is presented in Figure 4.6.

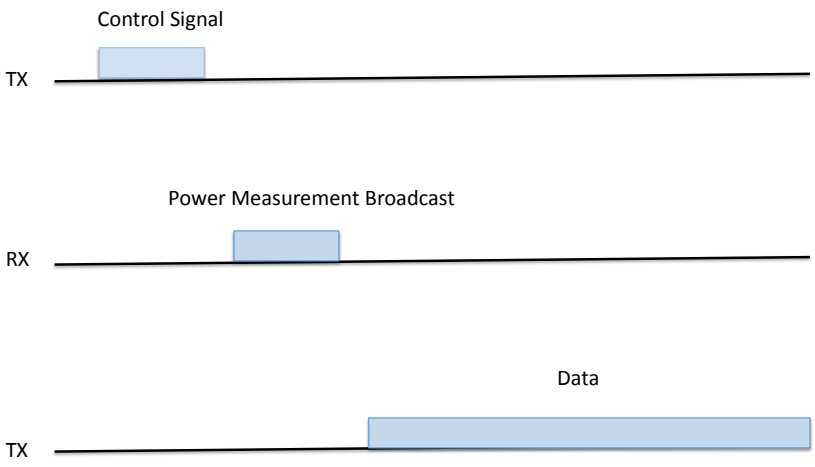

Figure 4.6: Signaling diagram in SINR based distributed backpressure algorithm

4.3.1 Simulation Model and Results

A 40-node uniform random topology in 5000 x 5000m plane is considered. The transmitter power of a node is 10 dBm. The path loss exponent is taken as 3 and the SINR threshold is 10 dB. The noise power is assumed to be -90 dBm. The rest of the simulation parameters remain the same as in the previous section. We compare our proposed algorithm with three centralized algorithms: Interference Graph, Power Control and Uniform Power. The last two algorithms were proposed in the chapter 3.

Figure 4.7 shows the average queue backlog performance of the aforementioned algorithms. The Interference Graph approach performs better than all other schemes. But the performance of our distributed algorithm is quite close to the Interference Graph approach. For most of the sample topologies taken, our distributed algorithm performs around 90% of the Interference Graph approach. An interesting observation is that our distributed algorithm does slightly better than uniform and power controlled approaches. The reason is that centralized approaches are conservative while evaluating the risk of interference.

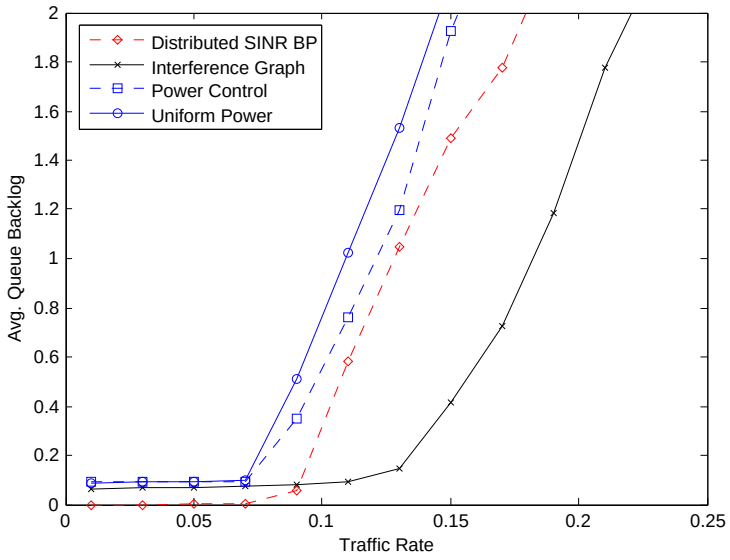

Figure 4.7: Average queue backlog performance-comparison of SINR based backpressure algorithms

4.4 Conclusion

We proposed distributed backpressure algorithms for one-hop and the SINR interference models. The distributed algorithms are built on the framework we developed in the previous chapters. The algorithms are fully distributed and requires only one-hop information. They also provide good competition to their corresponding centralized designs. The algorithms require minimum control signaling overhead and are simple to implement in realistic scenarios. Though it is clear that our distributed algorithms perform quite well, it would be interesting to simulate and analyze their performance in a standard real network simulator such as OPNET Modeler and OMNeT++. This would also help us to analyze the impact of signaling overhead present in the SINR based distributed backpressure algorithm. This is the goal of our future work.

Chapter 5

Future Directions and Conclusion

5.1 Multichannel Backpressure Algorithm

In previous chapters, we considered the single channel scenario — only one channel is present in the network and nodes that transmitting simultaneously interfere with each other if they are sufficiently close. Though single channel networks are common, multichannel networks are becoming increasingly popular, particularly with the advances in frequency-agile radios. Multi-channel multi-radio ad hoc wireless networks recently received a substantial amount of interest, especially under the context of wireless mesh neyworks [59]. Using multiple channels per node can increase the overall network throughput because interfering nodes can be assigned to different frequencies and can have simultaneous successful transmissions. But using multiple channels could be a restricted dynamic spectrum access scenario in which licensed users, also called primary users, have priority in spectrum access and the unlicensed or secondary users can only utilize unoccupied frequency bands. Since spectrum utilization among secondary nodes is contingent upon primary users, location and activity, different sets of frequencies available to use. In such cases, dealing with multichannel scenarios becomes a necessity. Such multi-channel multi-radio networks present a challenging resource allocation problem:

1. *Channel-assignment:* What are the set of channels that each link should operate on?

2. *Scheduling:* When should a link be activated on each channel?

3. *Routing:* How to select route for each source-destination pair that minimize interference and increase throughput?

A multichannel backpressure based algorithm can solve this problem. Some previous works address the issue of designing control protocols for multi-channel multi-radio ad hoc networks [60–66]. But either they are centralized designs or they are too complex to be implemented in real systems. This motivates our single-channel-based backpressure design to be adapted in multichannel networks.

We assume every node has multiple radios and multiple channels are present at every node with the possibility that nodes have different sets of available channels. For our design purposes, we assume that network topology has been formed initially. Though we assume that a network topology has already been initialized, possibly using so-called rendezvous techniques [67–69]. Every node utilizes all the radios it has, to maximize the number of neighbors. We can use generalized routing in this case. The generalized routing may use a combined metric of hop-count, channel switching delay, interference cost etc. Some examples of routing metrics in multi-channel case are given in [70, 71]. After running the generalized routing algorithm, the nodes update their routing tables with extended paths and shortest paths. To share queue information with neighbors, a node piggybacks the queue length information on data packets as described in Chapter 4. Based on the queue information, the nodes prefer shortest paths; if the queue length for any destination are larger than a predefined threshold, then extended paths are also utilized.

Unlike the single channel scenario, there is another dimension of optimization possible in multichannel backpressure algorithm: One can modify or adapt the topology in such a way that the backpressure algorithm can accommodate higher traffic load with new topology. For instance, the nodes can try to re-organize the network topology to provide high capacity routes for the congested flows. The topology adaptation can be performed by doing channel

assignment in distributed fashion. The CARD algorithm provides one example of such an optimization [72].

5.2 Conclusion

In our work, we have focused on a class of throughput-optimal network policies called backpressure policies. These policies use cross-layer optimization and do scheduling and routing jointly. Though these policies were proposed about twenty years ago, they are still not suitable for many practical applications, particularly for wireless ad hoc networks. In this work, we address three main problems with backpressure-based techniques. First, there is a design deficiency in backpressure routing which causes long and looping routes to be over-utilized and resulting in long delays. Second, many works on backpressure technique make some simplistic assumptions about scheduling and interference in wireless networks resulting in unrealistic scheduling decisions. Finally, it is challenging to implement backpressure policies in a practical, distributed manner.

The first issue is discussed in Chapter 2. We proposed a new technique for backpressure routing which address the problem of high end-to-end delay in routing. Our solution achieves nearly the same throughput region as the capacity region achieved by the optimal backpressure policy. Our packet forwarding scheme can be implemented with any generic routing protocol. In Chapter 3, we presented a novel technique that provides an SINR-based binary conflict graph. By using such conflict graph, the backpressure scheduling policy can utilize well-known graph-based scheduling algorithms while still satisfying realistic interference constraints. We finally presented distributed schemes in Chapter 4 that are based on our centralized designs. The distributed schemes require only one hop information and are easy to implement in the wireless ad hoc networks. We hope that our solutions will aid in future implementations of backpressure-based protocols in wireless ad hoc networks.

Bibliography

[1] L. Tassiulas and A. Ephremides, "Stability properties of constrained queueing systems and scheduling policies for maximum throughput in multihop radio networks," *IEEE Transactions on Automatic Control*, vol. 37, pp. 1936 –1948, December 1992.

[2] M. J. Neely, E. Modiano, and C. E. Rohrs, "Capacity and delay tradeoffs for ad-hoc mobile networks," in *Proceedings of the First International Conference on Broadband Networks (BROADNETS)*, 2004, pp. 428–438.

[3] L. Tassiulas and A. Ephremides, "Dynamic server allocation to parallel queues with randomly varying connectivity,," *IEEE Transactions on Information Theory*, vol. 39, pp. 466–478, December 1993.

[4] M. J. Neely, E. Modiano, and C. E. Rohrs, "Dynamic power allocation and routing for time varying wireless networks," *IEEE Journal on Selected Areas in Communications, Special Issue on Wireless Ad-hoc Networks*, vol. 23, pp. 89–103, January 2005.

[5] D. Shah and M. Kopikare, "Delay bounds for the approximate maximum weight matching algorithm for input queued switches," in *Proceedings of IEEE Conference on Computer Communications (INFOCOM)*, 2002, pp. 1024–1031.

[6] L. Tassiulas, "Scheduling and performance limits of networks with constantly changing topology," *IEEE Transactions on Information Theory*, vol. 43, pp. 1067–1073, May 1997.

[7] T. Moscibroda, "The worst-case capacity of wireless sensor networks," in *Proceedings of the 6th International Conference on Information Processing in Sensor Networks (IPSN)*, 2007, pp. 1–10.

[8] P. Chaporkar, K. Kar, and S. Sarkar, "Throughput guarantees through maximal scheduling in multihop wireless networks," in *Proceedings of 43rd Annual Allerton Conference on Communication, Control and Computing*, September 2005, pp. 28–30.

[9] J. Dai and B. Prabhakar, "The throughput of data switches with and without speedup," in *Proceedings of IEEE Conference on Computer Communications (INFOCOM)*, vol. 2, May 2000, pp. 556–564.

[10] X. Lin and N. Shroff, "The impact of imperfect scheduling on cross-layer rate control in multihop wireless networks," in *Proceedings of IEEE Conference on Computer Communications (INFOCOM)*, March 2005, pp. 1804–1814.

[11] X. Wu, R. Srikant, and J. R. Perkins, "Queue-length stability of maximal greedy schedules in wireless networks," in *Proceedings of Inaugural Workshop on Information Theory and Applications*, February 2006.

[12] A. Dimakis and J. Walrand, "Sufficient conditions for stability of longest queue first scheduling: Second order properties using fluid limits," *Journal on Advances of Applied Probability*, vol. 2, pp. 505–521, June 2006.

[13] C. Joo, "A local greedy scheduling scheme with provable performance guarantee," in *Proceedings of International Symposium on Mobile Ad-hoc Networking and Computing (MOBIHOC)*, 2008, pp. 111–120.

[14] C. Joo, X. Lin, and N. Shroff, "Understanding the capacity region of the greedy maximal scheduling algorithm in multi-hop wireless networks," in *Proceedings of IEEE Conference on Computer Communications (INFOCOM)*, April 2008, pp. 1103–1111.

[15] G. Sharma, R. R. Mazumdar, and N. B. Shroff, "On the complexity of scheduling in wireless networks," in *Proceedings of ACM International Symposium on Mobile Ad-hoc Networking and Computing (MOBICOM)*, 2006, pp. 227–238.

[16] X. Lin and S. Rasool, "Constant-time distributed scheduling policies for ad hoc wireless networks," in *Proceedings of 45th IEEE Conference on Decision and Control*, December 2006, pp. 1258–1263.

[17] A. Gupta, X. Lin, and R. Srikant, "Low-complexity distributed scheduling algorithms for wireless networks," *IEEE/ACM Transactions on Networking*, vol. 17, no. 6, pp. 1846–1859, 2009.

[18] X. Wu, R. Srikant, and J. R. Perkins, "Scheduling efficiency of distributed greedy scheduling algorithms in wireless networks," *IEEE Transactions on Mobile Computing*, vol. 6, no. 6, pp. 595–605, June 2007.

[19] S. Sanghavi, L. Bui, and R. Srikant, "Distributed link scheduling with constant overhead," *ACM SIGMETRICS Performance Evaluation Review*, vol. 35, no. 1, pp. 313–324, 2007.

[20] L. Tassiulas, "Linear complexity algorithms for maximum throughput in radio networks and input queued switches," in *Proceedings of IEEE Conference on Computer Communications (INFOCOM)*, vol. 2, Mar 1998, pp. 533–539.

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

[22] L. Jiang and J. Walrand, "A distributed CSMA algorithm for throughput and utility maximization in wireless networks," in *Proceedings of the Annual Allerton Conference on Communication, Control, and Computing*, September 2008, pp. 1511–1519.

[23] M. J. Neely, "Dynamic power allocation and routing in satellite and wireless networks with time varying channels," Ph.D. dissertation, Massachusetts Institute of Technology, 2003.

[24] M. Neely, E. Modiano, and C. Li, "Fairness and optimal stochastic control for heterogeneous networks," in *Proceedings of the IEEE Conference on Computer Communications (INFOCOM)*, vol. 3, March 2005, pp. 1723–1734.

[25] A. L. Stolyar, "Maximizing queueing network utility subject to stability: Greedy primal-dual algorithm," *Journal on Queueing System Theory Application*, vol. 50, no. 4, pp. 401–457, 2005.

[26] A. Dua and N. Bambos, "Distributed backlog-driven power control in wireless networking," in *Proceedings of IEEE Workshop on Local and Metropolitan Area Networks (LANMAN)*, June 2007, pp. 13–18.

[27] A. Giannoulis, K. P. Tsoukatos, and L. Tassiulas, "Maximum throughput power control in CDMA wireless networks," in *Proceedings of IEEE International Conference on Communications (ICC)*, vol. 10, June 2006, pp. 4457 –4462.

[28] Y. Xi and E. Yeh, "Throughput optimal distributed control of stochastic wireless networks," in *Proceedings of International Symposium on Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks (WiOPT)*, April 2006.

[29] M. J. Neely, "Energy optimal control for time-varying wireless networks," *IEEE Transactions on Information Theory*, vol. 52, no. 7, pp. 2915–2934, July 2006.

[30] S. Kompella, J. Wieselthier, and A. Ephremides, "Multi-hop routing and scheduling in wireless networks subject to SINR constraints," in *Proceedings of IEEE Conference on Decision and Control*, December 2007, pp. 5690–5695.

[31] B. Radunovic and J. L. Boudec, "Joint scheduling, power control and routing in symmetric, one-dimensional, multi-hop wireless networks," in *Proceedings of International*

Symposium on Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks (WiOPT), 2002.

[32] R. Bhatia and M. Kodialam, "On power efficient communication over multi-hop wireless networks: Joint routing, scheduling and power control," in *Proceedings of IEEE Conference on Computer Communications (INFOCOM)*, vol. 2, March 2004, pp. 1457–1466.

[33] R. Cruz and A. Santhanam, "Optimal routing, link scheduling and power control in multihop wireless networks," in *Proceedings of IEEE Conference on Computer Communications (INFOCOM)*, vol. 1, March 2003, pp. 702–711.

[34] M. Chiang, "To layer or not to layer: Balancing transport and physical layers in wireless multihop networks," in *Proceedings of IEEE Conference on Computer Communications (INFOCOM)*, vol. 4, March 2004, pp. 2525 – 2536.

[35] L. Georgiadis, M. J. Neely, and L. Tassiulas, "Resource allocation and cross-layer control in wireless networks," *Foundations and Trends in Networking*, vol. 1, no. 1, pp. 1–144, 2006.

[36] L. Ying, S. Shakkottai, and A. Reddy, "On combining shortest-path and back-pressure routing over multihop wireless networks," in *Proceedings of IEEE Conference on Computer Communications (INFOCOM)*, 2009.

[37] B. Radunović, C. Gkantsidis, D. Gunawardena, and P. Key, "Horizon: Balancing TCP over multiple paths in wireless mesh network," in *Proceedings of ACM International Conference on Mobile Computing and Networking (MOBICOM)*, 2008, pp. 247–258.

[38] A. Warrier, S. Ha, P. Wason, I. Rhee, and J. Kim, "DiffQ: Differential backlog congestion control for wireless multi-hop networks," in *Proceedings of Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks (SECON)*, June 2008, pp. 585–587.

[39] A. Sridharan, S. Moeller, and B. Krishnamachari, "Making distributed rate control using Lyapunov drifts a reality in wireless sensor networks," in *Proceedings of International Symposium on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks and Workshops (WiOPT)*, April 2008, pp. 452–461.

[40] A. Behzad and I. Rubin, "Impact of power control on the performance of ad hoc wireless networks," in *Proceedings of IEEE Conference on Computer Communications (INFOCOM)*, vol. 1, March 2005, pp. 102–113.

[41] T. ElBatt and A. Ephremides, "Joint scheduling and power control for wireless ad-hoc networks," in *Proceedings of IEEE Conference on Computer Communications (INFOCOM)*, vol. 2, 2002, pp. 976 – 984.

[42] K. Jain, J. Padhye, V. Padmanabhan, and L. Qiu, "Impact of interference on multi-hop wireless network performance," in *Proceedings of International Conference on Mobile Computing and Networking (MOBICOM)*, vol. 11, no. 4, 2005, pp. 471–487.

[43] T. Moscibroda, R. Wattenhofer, and A. Zollinger, "Topology control meets SINR: The scheduling complexity of arbitrary topologies," in *Proceedings of ACM International Symposium on Mobile Ad-hoc Networking and Computing (MOBIHOC)*, 2006, pp. 310–321.

[44] Y. Yi, de Veciana, and S. Shakkottai, "On optimal MAC scheduling with physical interference," in *Proceedings of IEEE Conference on Computer Communications (INFOCOM)*, May 2007, pp. 294–302.

[45] G. Brar, D. Blough, and P. Santi, "Computationally efficient scheduling with the physical interference model for throughput improvement in wireless mesh networks," in *Proceedings of ACM International Conference on Mobile Computing and Networking (MOBICOM)*, 2006, pp. 2–13.

[46] K. Leung, V. Li, and D. Yang, "An overview of packet reordering in transmission control protocol (TCP): Problems, solutions, and challenges," *IEEE Transaction on Parallel Distribution Systems*, vol. 18, no. 4, pp. 522–535, 2007.

[47] S. Kandula, D. Katabi, and S. Sinha, "Harnessing TCP's burstiness using flowlet switching," *Proceedings of ACM Workshop on Hot Topics in Networking (HotNets-III)*, November 2004.

[48] S. Kandula, D. Katabi, S. Sinha, and A. Berger, "Dynamic load balancing without packet reordering," *SIGCOMM Computer Communication Review*, vol. 37, no. 2, pp. 51–62, 2007.

[49] A. Tiwari and A. Sahoo, "A local coefficient based load sensitive routing protocol for providing QoS," in *Proceedings of International Conference on Parallel and Distributed Systems (ICPADS)*, 2006, pp. 447–456.

[50] J. O. Kim, T. Yamamoto, A. Yamaguchi, and S. Obana, "Adaptive multimedia flow splitting over WiMAX and WiFi links," *IEICE Transactions on Communications*, vol. 91-B, no. 10, pp. 3085–3094, 2008.

[51] W. Chao, Z. Xingming, C. Wenping, and N. Xiaona, "Load balancing algorithm using flow chopping to avoid packet reordering," in *Proceedings on International Forum on Information Technology and Applications*, vol. 2, 2009, pp. 193–197.

[52] U. K. Shukla and A. B. MacKenzie, "Delay sensitive backpressure routing over multihop wireless networks," *to be submitted*, 2010.

[53] L. Bui, R. Srikant, and A. Stolyar, "Novel architectures and algorithms for delay reduction in back-pressure scheduling and routing," in *Proceedings of IEEE Conference on Computer Communications (INFOCOM)*, 2009.

[54] A. Behzad and I. Rubin, "On the performance of graph-based scheduling algorithms for packet radio networks," in *Proceedings of IEEE Global Telecommunications Conference (GLOBECOM)*, vol. 6, December 2003, pp. 3432–3436.

[55] U. K. Shukla and A. B. MacKenzie, "Graph-based scheduling policies that meet SINR constraints in wireless networks," *to be submitted*, 2010.

[56] D. Chafekar, V. A. Kumar, M. Marathe, S. Parthasarathy, and A. Srinivasan, "Cross-layer latency minimization in wireless networks with SINR constraints," in *Proceedings of ACM International Symposium on Mobile Ad-hoc Networking and Computing (MOBICOM)*, 2007, pp. 110–119.

[57] O. Goussevskaia, Y. A. Oswald, and R. Wattenhofer, "Complexity in geometric SINR," in *Proceedings of ACM International Symposium on Mobile Ad-hoc Networking and Computing (MOBICOM)*, 2007, pp. 100–109.

[58] S. Sakai, M. Togasaki, and K. Yamazaki, "A note on greedy algorithm for maximum weight independent set problem," *Journal on Discrete Applied Mathematics*, 2003.

[59] P. Kyasanur, J. So, C. Chereddi, and N. Vaidya, "Multichannel mesh networks: Challenges and protocols," *IEEE Wireless Communications*, vol. 13, no. 2, pp. 30–36, april 2006.

[60] P. Kyasanur and N. H. Vaidya, "Capacity of multi-channel wireless networks: Impact of number of channels and interfaces," in *Proceedings of International Conference on Mobile Computing and Networking (MOBICOM)*, 2005, pp. 43–57.

[61] J. So and N. H. Vaidya, "Routing and channel assignment in multi-channel multi-hop wireless networks with single network interface," in *Proceedings of International Conference on Quality of Service in Heterogeneous Wired/Wireless Networks (QSHINE)*, August 2005.

[62] M. Alicherry, R. Bhatia, and L. Li, "Joint channel assignment and routing for throughput optimization in multi-radio wireless mesh networks," in *Proceedings of the International Conference on Mobile Computing and Networking (MOBICOM)*, 2005, pp. 58–72.

[63] A. Raniwala and T. Chiueh, "Architecture and algorithms for an IEEE 802.11-based multi-channel wireless mesh network," in *Proceedings of the IEEE Conference on Computer Communications (INFOCOM)*, vol. 3, march 2005, pp. 2223 – 2234.

[64] P. Bahl and R. Chandra, "SSCH: Slotted seeded channel hopping for capacity improvement in IEEE 802.11 ad-hoc wireless networks," in *Proceedings of the International Conference on Mobile Computing and Networking (MOBICOM)*, 2004, pp. 216–230.

[65] X. Lin and S. Rasool, "A distributed joint channel-assignment, scheduling and routing algorithm for multi-channel ad-hoc wireless networks," in *Proceedings of the IEEE Conference on Computer Communications (INFOCOM)*, May 2007, pp. 1118–1126.

[66] S. Merlin, N. Vaidya, and M. Zorzi, "Resource allocation in multi-radio multi-channel multi-hop wireless networks," in *Proceedings of the IEEE Conference on Computer Communications (INFOCOM)*, April 2008, pp. 610–618.

[67] J. Mo, H. S. Wilson, and J. Walrand, "Comparison of multi-channel MAC protocols," in *Proceedings of ACM International Symposium on Modeling, Analysis and Simulation of Wireless and Mobile Systems (MSWiM)*, 2005, pp. 209–218.

[68] A. Tzamaloukas and J. J. Garcia-Luna-Aceves, "Channel-hopping multiple access," in *IEEE International Conference on Communications (ICC)*, vol. 1, 2000, pp. 415–419.

[69] A. T. Garcia-Luna-Aceves and J. J. Garcia-Luna-Aceves, "Channel hopping multiple access with packet trains for ad-hoc networks," in *Proceedings of IEEE Mobile Multimedia Communications (MoMuC)*, 2000.

[70] Z. Yang, G. Cheng, W. Liu, W. Yuan, and W. Cheng, "Local coordination based routing and spectrum assignment in multi-hop cognitive radio networks," *Journal of Mobile Network Applications*, vol. 13, no. 1-2, pp. 67–81, 2008.

[71] G. Cheng, W. Liu, Y. Li, and W. Cheng, "Spectrum aware on-demand routing in cognitive radio networks," in *Proceedings of IEEE International Symposium on New Frontiers in Dynamic Spectrum Access Networks (DYSPAN)*, April 2007, pp. 571–574.

[72] R. Irwin and L. DaSilva, "Channel assignment based on routing decisions (CARD): Traffic-dependent topology control for multi-channel networks," in *Proceedings of IEEE International Conference on Communications (ICC) Workshops*, June 2009.