OPTIMIZING TRAFFIC DISTRIBUTION IN MULTI-RADIO MULTI-CHANNEL WIRELESS MESH NETWORKS UNDER DYNAMIC TRAFFIC DEMAND

By

Liang Dai

Dissertation

Submitted to the Faculty of the

Graduate School of Vanderbilt University

in partial fulfillment of the requirements

for the degree of

DOCTOR OF PHILOSOPHY

in

Computer Science

August 2011

Nashville, Tennessee

Approved:

Professor Yuan Xue

Professor Yi Cui

Professor Larry Dowdy

Professor Douglas C. Schmidt

Professor Bradley A. Malin

To my parents
and
anyone who has been part of my life

ACKNOWLEDGEMENTS

First, I would like to thank everyone who has helped me in my life, no matter whether it was when I was still a child or during my graduate study. I would not be who I am without help from all of you. I would like to especially thank my parents. I could not make it without their endless support for my life. I know I owe them more than I could repay.

This work would not have been possible without the guidance and support of my advisor, Dr. Yuan Xue. I would like to thank her for her patience, time, advice and invaluable support throughout this study. I am grateful to the members of my dissertation committee, Dr. Yi Cui, Dr. Larry Dowdy, Dr. Douglas C. Schmidt, and Dr. Bradley A. Malin for their insight and guidance.

Finally, I would like to thank all past and current members of VANETS lab. It has been my pleasure to study and collaborate with all of you. Thank you so much for the help you have given me. In addition, I would to like thank all my friends at Vandy. You have made my experience at Vandy wonderful.

TABLE OF CONTENTS

LIST OF TABLES

Table | Page

LIST OF FIGURES

CHAPTER I

INTRODUCTION

Wireless Mesh Networks

Wireless mesh networks (WMNs) have attracted increasing attention by research groups as well as for-profit companies [1, 2, 3, 4, 5, 6, 7]. Their easy-to-deploy feature and stable structure make them a competitive candidate as a high-performance and low-cost solution to last-mile broadband Internet access and disaster recovery in areas where wireline network deployment is impossible or expensive.

A WMN usually consists of local access points, relaying wireless routers, and gateways. Local access points aggregate the traffic from associated mobile clients. They communicate with each other and with relaying wireless routers, forming a multi-hop wireless backbone network that forwards the traffic to gateways, which have wireline connections to the Internet. In this dissertation, we refer to local access points, gateway access points, and mesh routers all as mesh nodes.

As a multi-hop wireless network, the WMN is easy to deploy. In areas where deploying cables is expensive or impossible, WMNs provide a practical solution for broadband Internet access with reduced deployment cost. In addition, WMNs have the following features that make them different from other multi-hop wireless networks:

- *Fixed Topology.* Compared with mobile ad hoc networks, where nodes are mobile and topologies are dynamic, mesh nodes in WMNs are usually fixed at specific locations. Since WMNs usually have infrequent topology changes [8] and relatively stable backbone structures, their routing paths are usually maintained for longer periods of time to reduce the route control message overhead. Identifying the high throughput paths is thus particularly important for WMNs. In mobile ad hoc networks, on the other hand, maintaining network connectivity is the key issue for routing.

- *Consistent Power Supply.* Fixed node location in WMNs also makes it possible for mesh nodes to use stable power supplies from external systems. Compared with wireless sensor networks, which operate on batteries with limited power supply, WMNs face fewer constraints on energy consumption in their design.

- *Multiple radio interfaces.* To improve the network capacity, the mesh nodes are usually equipped with multiple radio interfaces which can operate over different frequency spectrums simultaneously. This type of mesh network is usually referred to as a *multi-radio multi-channel* mesh network. The availability of multiple ratios provides a new dimension for traffic load balancing (*i.e.*, along the spectrum domain). Thus, it has the potential to enhance the network capacity, but it also complicates the design of traffic distribution.

Problem Description and Research Goal

Traffic distribution plays a critical role in determining WMN performance. The traffic collected at local access points needs to be delivered via multiple relaying routers to reach the gateways that are connected to the Internet and vice versa. In a multi-radio multi-channel mesh network, traffic distribution involves two major components: channel assignment and traffic routing. Wireless communication standards, such as IEEE 802.11 [9], divides frequency bands into channels. This allows simultaneous transmissions over different channels whose frequency bands do not overlap. The purpose of channel assignment is to assign a radio interface to operate on a channel during a certain time interval. Once radio interfaces are assigned to channels, it determines the set of neighbors that a node can communicate directly with, as two nodes can communicate only if they have radio interfaces over the same channel. This also means that the network topology is determined for this time interval. On the other hand, traffic routing distributes the traffic along different paths (in the spatial domain) and different channels (in the spectrum domain) in the network. For a wireless relaying node with multiple interfaces, traffic routing determines which radio interface an incoming-flow (or a packet) should be forwarded to and which neighboring node it should be sent to.

While channel assignment is a new problem for wireless networks, traffic routing is a classic problem that has been extensively studied in wireline networks [10, 11, 12, 13]. However, the routing algorithms introduced in the context of wireline networks cannot be applied to WMNs directly due to the following unique features of WMNs:

- *Wireless Interference.* Wireless communication suffers from location dependent interference. A variety of models (*e.g.*, physical model [14], protocol model [14]) has been developed to characterize such interference effects at different levels in the existing literature. To provide a simple overview, if two radio interfaces on the same channel are within the interference range of each other, then they cannot transmit at the same time. On the other hand, if they are far away from each other (*i.e.*, out of the interference range), they can transmit simultaneously, which is called spatial reuse of the channel. The location dependent resource contention couples the available bandwidth of a wireless link with the media access control and link-level scheduling scheme, and complicates the traffic distribution problem.

- *Availability of Multiple Radios and Multiple Channels.* While the availability of multiple radios and multiple channels in a WMN has the potential to improve the network performance, effectively utilizing them presents two new dimensions to the traffic distribution problem. Radio-to-channel assignment and traffic-to-radio forwarding considered under the wireless interference constraint are challenges that do not exist in wireline networks.

- *Dynamic Traffic.* The traffic in WMNs is highly dynamic and bursty, compared with the wireline peers. This is mainly due to the mobility of clients and the insufficient traffic multiplexing at local access points. As a result, the traffic pattern observed at WMNs is significantly different from the Internet backbone network, which requires new traffic models.

Given the importance of traffic distribution to the performance of WMN, it has recently become a research focus. Several algorithms have been proposed for traffic routing in single-radio and single-channel mesh networks, as well as joint routing and channel assignment in multi-radio and multi-channel mesh networks. The proposed approaches usually fall in two ends of the spectrum. On one end of the spectrum are the heuristic routing algorithms (*e.g.*, [15, 16, 17]). Although many of them are adaptive to the dynamic environments of wireless networks, these algorithms lack the theoretical foundation to allow analysis of how well the network performs globally (*e.g.*, whether the traffic shares the network in a fair fashion).

On the other end of the spectrum, there are theoretical studies that formulate mesh network routing as optimization problems (*e.g.*, [18, 19]). The routing algorithms derived from these optimization formulations can usually claim analytical properties such as resource utilization optimality and throughput fairness. In these optimization frameworks, traffic demand is usually implicitly assumed as static and known a priori. Contradictorily, recent studies of wireless network traces [20] show that the traffic demand, even being aggregated at access points, is highly dynamic and hard to estimate. Such observations have significantly challenged the practicability of the existing optimization-based routing solutions in WMNs.

The objective of this dissertation is *to design and evaluate an optimization-based traffic distribution solution for multi-radio, multi-channel wireless mesh networks which takes into account the dynamic nature of wireless traffic demand.*

Research Approach and Dissertation Contribution

In this dissertation, we propose a framework that integrates traffic distribution with traffic prediction as shown in Fig I.1. The traffic analysis component of the framework establishes traffic models for wireless access points and uses them to predict the future traffic demand for load distribution. In particular, we characterize historical traffic using time series models, then predict the future traffic demand based on the established models. We propose two different traffic prediction models. The single value prediction provides the expected value for predicted traffic demand. The prediction with statistic distribution provides possible traffic values with their corresponding probabilities. These two traffic prediction models will be used in two different routing algorithms in the traffic distribution component.

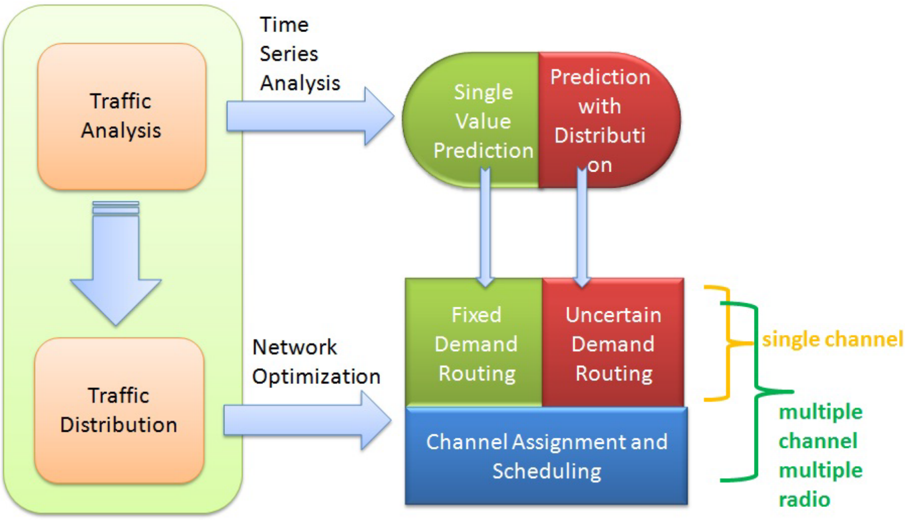


Figure I.1: Framework Overview

In the traffic distribution component, we develop optimization-based algorithms to balance the traffic load. The fixed-demand routing algorithm takes the single-value prediction result and computes the optimal routing based on the determined traffic information. The uncertain-demand routing takes the result from the prediction with distribution traffic model as the input and computes the routing paths that optimize the expected network performance. By considering the distribution information, we can minimize the impact of sub-optimal routing solutions due to the prediction error caused by the single-value traffic prediction. We study our routing algorithms under three different network models. The first network model has only a single channel. The routing algorithms developed under this model serve as the baseline algorithms for the other two network models. The second network model incorporates the existence of multiple radios and multiple channels, where the baseline routing algorithms in the first network model are extended to joint solution of channel assignment and routing solutions. The third network model considers the wireless random losses in traffic distribution.

The original contributions of this dissertation as follows:

- Practically, the integration of traffic estimation and distribution optimization effectively improves the performance of WMNs under dynamic and uncertain traffic. Most existing routing algorithms ignore the fact of dynamic traffic, which can lead to poor results when implemented in real networks. Our algorithms are adaptive to dynamic traffic and can improve wireless network performance under uncertain traffic. The full-fledged simulation study based on real wireless network traffic traces provides convincing validation of the practicability of our solution.

- Theoretically, we extend the classical linear network optimization algorithm, which only accepts the fixed-value demand as input, into a stochastic optimization solution capable of serving uncertain demands that are modeled by their statistical distributions.

Dissertation Organization

The remainder of this dissertation is organized as follows: Chapter II presents an overview of existing related literature, which includes traffic routing in wireline networks, wireless network routing and network traffic analysis. Chapter III introduces WMN models. Chapter IV describes how we use network traces to characterize traffic and explains the methods we take to predict traffic. In Chapter V, we present our formulation of the WMN routing problem and propose routing algorithms that are capable of handling dynamic traffic input for single channel WMNs. Chapter VI presents joint channel assignment and routing algorithms, which extend routing algorithms proposed in Chapter V to multi-radio and multi-channel WMN environment. Algorithms presented in Chapter V and VI assume ideal stable wireless connections. In Chapter VII, we extend solutions for lossy WMNs, where packet loss exists in wireless transmission. Finally, Chapter VIII concludes this dissertation.

CHAPTER II

BACKGROUND AND RELATED WORKS

This dissertation is related to work from three areas: 1) network routing, 2) traffic analysis, and 3) optimization methods with uncertain input. This chapter will review these existing works and highlight the open issues that remain unaddressed.

Network Routing

Routing refers to the operation of choosing paths from the source node to the destination nodes for packet delivery. It balances the traffic load throughout the network and plays a critical role in determining the network performance. A well-designed routing algorithm, depending on its specific goal, can increase network throughput, decrease network congestion, and/or minimize packet delays.

Routing Algorithms for Wireline Networks

Routing in wireline networks has been investigated extensively. The existing routing algorithms can be broadly classified into two categories: the single-flow routing algorithm and the multiple-commodity routing algorithm.

The single-flow routing algorithm considers the performance of a single flow independently as its objective. The routing objective is usually defined using link cost metrics, and the goal is to choose a path with minimum cost for a given flow. The routing algorithms are usually designed based on Dijkstra's algorithm [10, 11, 12, 13, 21, 22] in a centralized manner or Bellman-Ford algorithm in a distributed manner [23, 24, 25].

A communication network usually needs to support multiple flows between different source and destination nodes. Optimizing the routing paths for individual flows separately may lead to sub-optimal network performance if viewed globally. For example, some low-cost links may get overloaded by traffic. The multi-commodity routing algorithms are proposed to address this problem and optimize the performance of all flows simultaneously in the network. These algorithms usually formulate routing problems as optimization problems and compute routing paths according to the solution of the formulation. Depending on the optimization objective, the optimization-based routing algorithms include minimum cost routing [26, 27, 28], maximum throughput routing [29, 30] and maximum concurrent flow [31, 32]. The minimum cost routing is to find a set of paths with minimum aggregated cost. The maximum throughput routing algorithms maximize the aggregated throughput of all flows. Usually maximum throughput routing algorithms do not consider the demand of each flow, and the routing paths computed by these algorithms may be unfair to some flows. The maximum concurrent flow routing algorithm solves this problem by including traffic demand in the routing formulation so that the algorithm finds optimal paths based on the demand scaling factor of all flows, which is the ratio of the routed throughput and the traffic demand of the flows. For this reason, the maximum concurrent flow routing algorithm is the best fit for balancing traffic distribution in a network.

Routing algorithms for wireline networks provide a solid theoretical foundation for designing WMN routing algorithms. For example, many heuristic algorithms for WMNs use link metrics as a criteria for routing. They are similar to the single-flow routing algorithm in wireline networks. The unique features of wireless transmission, however, prevent the direct application of existing wireline network routing algorithms to WMNs. For example, several research results [33, 34, 16] show that the shortest-flow routing algorithm has to consider wireless losses and other wireless features in its cost metrics definition in order to be applied to WMNs. In fact, routing algorithms based on minimum hop count may lead to poor network performance because they favor more distant links, which may cause larger interference that decreases network capacity. Optimization based routing algorithms for WMNs can also find their counterparts in multiple-commodity routing algorithms that are designed for wireline networks with adaptation to WMN characteristics, including wireless interference, scheduling and radio to channel allocation, etc. The routing algorithms for wireless networks are reviewed in the next section.

Wireless Network Routing

The existing routing algorithms for WMNs fall at two ends of the spectrum: they are either heuristic or optimization-based. Heuristic algorithms are usually adaptive to dynamic traffic environments. However, they lack the theoretical foundation to analyze the throughput a WMN could achieve. Optimization based approaches, on the other hand, can usually claim resource utilization optimality, but most of those algorithms assume static traffic. The routing algorithms can also be classified as distributed and centralized algorithms based on the protocol implementations. Distributed algorithms (*e.g.*, [17, 35, 36]) perform the routing decisions locally based on the information received from neighboring nodes. They make the system scalable and easy to manage, but often suffer from sub-optimal performance penalty. Centralized algorithms (*e.g.*, [37, 38, 18]), by managing the traffic routing globally, usually outperform the distributed algorithms. However, overhead introduced by control messages can make centralized algorithms less efficient and harder to manage compared to the distributed ones.

Table II provides an overview of the existing routing algorithms. As evident in this table, this dissertation provides an important piece of information that has been missing in previous work.

		Adaptive to Network Traffic	**Not adaptive to Network Traffic**
Global Network Performance Assurance	**Yes**	The work of this dissertation	Optimization-based Algorithms
	No	Heuristic Algorithms (ETT, CARP, ...)	Heuristic Algorithms (ETX, ETOP, ...)

Table II.1: Existing Routing Algorithms for WMNs

Heuristic Algorithms

One feature of heuristic algorithms is their adaptiveness to dynamic environments of wireless networks. Although there are some centralized heuristic algorithms, the majority are performed in a distributed way, which makes the algorithms scalable. Those algorithms put different strategies into mesh nodes and attempt global optimization by optimizing at each local node. Nodes in most distributed algorithms need to collect information from their neighbors, make decisions based on the information received, and perform changes according to predefined protocols. Because each node make its own decision, the systems are agile for configuration changes and easy to implement. The disadvantage of heuristic algorithms is that they lack the theoretical foundation to analyze how well the network performs globally. It is still unknown whether the network resource is fully utilized.

The approaches of heuristic algorithms vary greatly from one to another. One popular approach is to use a routing metric to find optimal single-flow routing paths. A metric can be defined in several ways, including loss rate [33], transmission time [15], interference [39] and link congestion [40]. The definition of the routing metric is important because it can directly influence the algorithm performance.

Routing Metrics

Due to wireless transmission and signal interference, the traditional shortest path routing algorithms in wired networks, also known as minimum hop count routing, are no longer the best routing algorithms for WMNs. One possible approach to overcome this deficiency is to revise the routing metric, which is the hop number in minimum hop count routing, so that paths found by new routing algorithms can reduce the interference and increase the throughput. A new metric can consider either low level features, such as link lost rate, or high level status, such as flow rate and channel configuration, or both, depending on which WMN scenario is addressed. A common method of implementing metric-based algorithms is to collect and exchange the link cost information from neighboring nodes. Given this information, routing algorithms find the path based on the link cost metrics. Below, we review some metrics used in the existing heuristic routing algorithms.

ETX (Expected Transmission Count) [33]: Link loss is a common phenomenon in wireless networks, yet simply choosing a link based on a loss rate threshold may be insufficient. The ETX metric is an early work to address this problem. It considers the effects of link loss, asymmetry of loss ratios between two directions of each link and the interference among the successive links of a path together. The primary goal of ETX design is to find paths with high throughput despite losses.

In the formulation, the expected number of transmissions is expressed as follows: $ETX = \dfrac{1}{d_f \times d_r}$ where d_f is the forward delivery ratio and d_r is the reverse delivery ratio. The delivery ratios d_f and d_r are measured by using dedicated link probe packets. Each probe packet is broadcasted at an average period τ and no acknowledgement and retransmission are involved. Every node records the probes it receives during the last w seconds and calculates the delivery ratio from the sender at time t as : $r(t) = \dfrac{count(t-w,t)}{w/\tau}$, where $count(t-w,t)$ is the number of packets received during the last w seconds and w/τ is the number of packets expected to be received from the sender. Each probe sent by node X carries the number of packets received by X from each of its neighbors during the last w seconds, which allows neighboring nodes to calculate the delivery ratio to node X. The ETX of a route is the sum of the ETXs of links.

ETX addresses the throughput issue by choosing the delivery ratios, and it handles asymmetry by incorporating loss ratios in both directions. One of the major reasons that ETX does not use the product of link delivery ratios as the route metric is that the product fails to account for inter-hop interference. Instead, it uses the sum of link delivery ratios to penalize routes with more hops, which have lower throughput.

However, the problem with ETX is that it does not consider link capacity and congestion. It does not balance the traffic to avoid congested links. Thus it may not fully utilize the network resource or balance the network traffic. Also, ETX was developed at an earlier time when multi-radio had not been not introduced to increase network capacity, so ETX does not include the multi-radio feature in the metric. This is a drawback of ETX because the multi-radio feature has become common in wireless networks.

WCETT (Weighted Cumulative Expected Transmission Time) [15]: WCETT is an improvement of ETX. It includes not only the loss rate, which ETX also does, but also the link bandwidth. WCETT is a metric describing a whole route, which is comprised of several individual links, while Expected Transmission Time (ETT) only describes an individual link weight. As an improvement of ETX, WCETT incorporates both the link capacity and the multiple channels into existing solutions to make better use of available network resources.

The ETT of a link is defined as a "bandwidth-adjusted ETX" and is expressed as $ETT = ETX * \frac{S}{B}$ where S is the size of packets and B is the bandwidth of a link. The proposed WCETT is calculated as follows: $WCETT = (1 - \beta) \times \sum_{i=1}^{n} ETT_i + \beta \times \max_{1 \leq j \leq k} X_j$, where X_j is the sum of transmission times (ETT) of hops on channel j and β is a tunable parameter subject to $0 \leq \beta \leq 1$. The use of X_j is to encourage channel diversity. Finding $\max X_j$ is to search the most congested channel set. Based on the definition, the WCETT metric tries to balance between the global and local network performance. The first term reflects the total network resource consumed on this path while the second term reflects the channel set that has the most impact on the throughput. At the same time, WCETT is also a combination of delay and throughput where the first term is a measurement of path latency and the second term is path throughput.

Although WCETT can claim better network performance than ETX, there are still some remaining issues. One of the issues is that WCETT does not consider the traffic demand. Although the link capacity is included in the metric, the actual available/consumed bandwidth is not considered, which may cause unbalanced traffic load and network congestion. Although WCETT also considers the multi-radio feature, it does not assign channels to links, thus it depends on the pre-assigned channel schemes to find paths. It is difficult to reach the global network optimum without considering channel assignment in the metric. Finally, WCETT does not capture the inter-flow interference.

iAWARE (Interference Aware) [39]: iAWARE has a similar structure as WCETT, but as an improvement, iAWARE also considers the flow interference. In both ETX and WCETT, only intra-path interference is captured, and paths with more hops are less likely to be selected due to the interference. However, these metrics do not consider the interference effect among different paths. iAWARE, on the other hand, captures the effects of variation in link loss-ratio, differences in transmission rates, as well as inter-path and intra-path interference.

In iAWARE, the physical interference model (described in Chapter III) is introduced. In this model, the signal-to-noise ratio is used to measure how signals from other nodes interfere at receiving nodes. The interference ratio $IR_i(u)$ for a node u in a link $i = (u, v)$ is defined as the ratio of noise received over total interfered signal, including the noise and weighted signal interference from other nodes, and the interference ratio IR_i for link $i = (u, v)$ is $\min(IR_i(u), IR_i(v))$. When there is no interference, IR_i is 1 and the link quality is only determined by the link loss-ratio and the data rate, which are captured by ETT. The new metric $iAWARE(p)$ for a path p is defined as follows: $iAWARE(p) = (1-\alpha) \times \sum_{i=1}^{n} iAWARE_i + \alpha \times \max_{1 \leq j \leq k} X_j$, where $iAWARE_j$ for link j is defined as $iAWARE_j = \dfrac{ETT_j}{IR_j}$. X_j has a similar definition as that in ETT, which exploits the channel diversity. According to the definition of link metric $iAWARE_j$, IR_j is the wireless interference and ETT_j captures the link loss and retransmission. A higher IR_j and a lower ETT_j produce a lower $iAWARE_j$, which means better link quality.

Although iAWARE can be applied to network scenarios with multiple radios, it cannot incorporate this feature into the metric, and thus, like WCETT, it is unable to optimize routing together with channel assignment. Also, traffic balancing is not included in iAWARE, which could lead to poor performance when a network is congested.

ETOP (Expected Number of Transmissions On a Path) [41]: All of the previous described metrics assume that the number of retransmissions is unlimited. In reality, when the link quality is low and the transmission fails, a retransmission will always be initiated until packet delivery is successful. However, this assumption does not hold in real network scenarios. Instead, only a limited number of retransmissions are performed before a packet is discarded and a transmission failure is reported to the source. End-to-end transmission, depending on different applications, usually will be initiated from the source node instead of the intermediate node. The fact that the number of retransmissions is limited makes paths with weak links (high loss rate) that are closer to the destination unfavorable. When the transmission at a weak link fails and the retransmission is initiated from the source, fewer retransmissions are required when the weak link is closer to the source node. The metric of ETOP is based on the number of retransmissions, the link loss rate, as well as the link position in the whole path. It provides a new view on the link stability problem for wireless mesh routing. It shows how the position of a weak link in the path can affect the routing, which is a closer formulation to real networks. However, similar to other metrics, ETOP fails to include traffic balancing and channel assignment in the metric. Thus, traffic congestion and sub-optimal channel assignment still remain as open issues.

There are other proposed metrics that consider several features in WMNs, such as wireless interference and link loss rates. For example, ENT (Effective Number of Transmission) [42] is a quality-aware routing metric. The premise for such a heuristic is that the loss rate of a link can vary from time to time. Using a mean value to measure the link loss rate may be inaccurate and lead to poor performance because it fails to adapt to burst loss conditions. The ETX metric cannot update the link loss status frequently due to the overhead. In contrast, ENT provides a framework that combines the mean and standard deviations of link loss ratios to capture the time-varying characteristic of wireless links.

The link loss ratio is not the only metric that can reflect link quality. The link capacity and available bandwidth are also important metrics to a link. For example, CARP (Channel Characteristics-Aware Routing Protocol) [43] uses residual link bandwidth to balance the traffic load over a network and avoid link congestion. In addition, the CARP based routing algorithm supports multi-path routing in order to exploit potential throughput. As another example, CCM (Channel Cost Metric) [35] takes both interference and channel diversity into account and reflects the interference cost. Unlike previous metrics, a distributed channel assignment algorithm works together with CCM to assign channels to links in a distributed fashion, which is an improvement to the previous metrics where channel assignment was assumed to be performed before routing. Furthermore, most link metrics assume that all links use the same link rate for transmission. However, wireless links may select a different rate from a set of rates predefined by MAC protocols. Different rates may, in turn, affect other important factors, including link loss rate and transmission distances. In [44], the relationship between different rates, transmission distance, and other factors are studied. They proposed a new metric that considers transmission rates and show that it tends to find paths with higher throughput. Additionally, multicast routing metrics for WMNs are discussed in [45],

The link quality is included partially or entirely by link metrics. One of the challenges of implementing a metrics based routing algorithm is how to measure the link quality accurately. Inaccurate measurement of the link quality can cause a routing algorithm to choose suboptimal relaying nodes and low-quality paths. In [46], the authors discuss pros and cons of existing techniques for measuring the link quality. They point out that measurement techniques should consider both accuracy and efficiency. The broadcast-based active probing, used by ETX and ETT, is inexpensive. However, since it uses a fixed and low data rate, it is more tolerant of errors, which can be more optimal than the actual link loss rate. They also point out that measurement techniques should be aware of link asymmetry and be sufficiently flexible to cope with the time-varying link quality without introducing extra overhead.

Designing a good routing metric requires consideration of several aspects of WMNs to ensure that the algorithm is valid and efficient. In [47], several requirements for evaluating a metric are discussed. For example, proposed metrics should be loop free and produce good routing performance, and an algorithm for finding paths based on metrics should be efficient. Another important requirement for a routing-based algorithm is that routing metrics should ensure route stability. Additionally, most routing metrics discover routing paths hop-by-hop based on metrics without global coordination. Once the paths are found, they may not be updated or may be updated less frequently. There are two reasons behind such a design: 1) the metric itself does not change frequently; 2) frequent route changes will result in routing instability, introduce operation overhead and degrade network performance. In [48], routing stability is studied on two actual networks using the WCETT protocol described above. Their findings show that dominated routes are short-lived because of an excessive number of route flaps, and the routing metric should balance the performance adaptability and routing stability.

The major limitation of metric-based routing algorithms is that they lack a theoretical foundation for global resource coordination. For example, most metric-based algorithms described above cannot handle the channel assignment issue for multi-radio WMNs. They can diversify the channel selection for routing based on a preassigned channel scheme, but there is no perfect solution for these algorithms to participate in channel assignment to optimize the whole network. Some algorithms consider the link capacity in their metrics, but they fail to balance the traffic and can overload links with large capacities. These are limitations that cannot be addressed by link metrics themselves. As an alternative solution, a system-wide optimization needs to be applied for routing to coordinate constraints and reach a global network performance optimum.

Other Heuristic Algorithms

In addition to link-metric-based routing, several other heuristic routing, and joint routing, and channel allocation algorithms are presented in the existing literature. In [17], the authors propose a distributed algorithm similar to BGP (Border Gateway Protocol), where reachability information is broadcasted when a routing tree is constructed. Nodes are assigned with priorities according to their local environments. There are several works (*e.g.*, [49, 50]) in particular that focus on the channel assignment. [49], for instance, proposes a dynamic interference-aware channel assignment algorithm. As another example, the channel assignment algorithm in [50] considers the interference problem among orthogonal channels due to crosstalk or leakage. Additionally, Slotted Seeded Channel Hopping (SSCH) [51] is a hop-by-hop routing algorithm that uses time multiplexing to exploit the channel diversity. The goal of this algorithm is to design a channel switching scheme so that nodes that want to communicate can be on the same channel while avoiding the use of overlapping channels for the interfered links.

Opportunistic Routing

In wireless networks, the connectivity between two nodes can be intermittent. In the traditional routing schemes presented above, packets are usually forwarded along a fixed path which is determined by the routing algorithm. Opportunistic routing, on the other hand, forwards the packets in an undeterministic way by choosing among multiple forwarders [16, 52] and multiple paths [53, 54], depending on which nodes actually receive the packets. By using high risk resources (*e.g.,* high loss-rate links), it [16, 55, 53] has been shown that opportunistic routing algorithms can have better throughput than traditional routing.

GeRaF [56] is an earlier work studying opportunistic routing algorithms. It assumes that each node in a network has precise location information of other nodes. Unlike later studies, the forwarding nodes exploit the next relaying node by themselves and then pick nodes close to the destination to which they forward packets. An improved algorithm, ExOR [16], is an integrated routing-and-MAC-opportunistic routing algorithm. ExOR broadcasts the message to neighboring nodes with an ordered list of preferred relaying nodes. The priority order is based on the expected cost, a metric similar to ETX, of delivering a packet from each node in the list to the destination. A node will wait its turn to forward the message unless it is notified that nodes before it in the list already did. At the same time, ExOR operates on batches of packets. When 90% of the batch messages have been forwarded through the priority list, the remaining packets are forwarded through traditional routing to control the routing cost. One of the limitations of ExOR is that it prevents spacial reuse and thus underutilizes the wireless medium. MORE [53] improves ExOR with intra-flow network coding. It mixes the packets randomly before forwarding to ensure that routers receiving the same transmission do not forward the same packet. The work of [55] further extends MORE with the consideration of multiple data transmission rates in wireless networks.

Optimal Routing Algorithms

Though empirical studies have validated the performance of many heuristic wireless mesh routing algorithms under different network scenarios, the heuristic designs lack the theoretical foundation to generalize the empirical results and analyze the network performance with respect to the corresponding optimal cases. On the other hand, there are theoretical studies that formulate mesh network routing as optimization problems (*e.g.*, [18, 38, 19]). The routing algorithms derived from this optimization formulations often claim analytical properties, such as resource utilization optimality and throughput fairness.

There are multiple factors affecting the performance of a network, and choosing the right model to describe the network is an important step that affects the formulation for the optimization problem. Common wireless interference models include the *protocol model* and the *physical model* (Chapter III). A popular approach is to use the *protocol model*, which is simpler than the *physical model*. The interference in the *physical model* is based on actual signal-to-noise ratio, which may change from one time to another depending on the environment. This makes problem formulation difficult because each node has its unique way of computing the interference. The *protocol model*, on the other hand, uses the concept of a precomputable *interference set* to represent wireless interference to simplify the computation.

A common approach for optimal routing formulation is based on linear programming (LP). The objective of LP is to maximize the throughput or minimize the most congested region, which is similar to the maximum flow problem in wireline networks. However, there are some differences between the problem in WMNs and the one in wireline networks. One of the major differences is that the link capacity constraint is the only constraint in wireline networks, while in WMNs, the wireless interference also limits performance. Thus the constraints on LP models usually include the link capacity, interference set capacity (in *protocol model*), and flow requirements.

The solution space and the flow requirements vary in different methods proposed for WMN optimization. For example, [18] and [38] investigate the optimal solution of joint channel assignment and routing for maximum throughput under a multi-commodity flow problem formulation. As another example, [19] presents bandwidth allocation schemes to achieve maximum throughput and lexicographical max-min fairness. Also, distributed algorithms have been proposed for joint scheduling and routing as well as for joint channel assignment, scheduling, and routing in [36]. The distributed algorithms use local information for traffic routing and thus have the potential to accommodate dynamic traffic.

LP-based routing optimization algorithms manage the system resources globally and can produce close to optimal values. LP models assume that mesh nodes should have similar or the same configurations, which makes it simple to analyze the problem. For example, LP models usually assume that each node in a WMN has the same transmission and interference range, and apply the *protocol model* in the system. However, [57] shows that actual values of interference and transmission ranges are more periodic with several values instead of a single value, and these can vary across nodes. Another problem with optimization based algorithms is their flexibility. Their solutions are based on global information and may not be agile enough to adapt according to individual wireless transmission changes. Finally, optimization-based algorithms usually require global information to perform routing, which makes it difficult to deploy in real systems.

Solving the routing optimization usually requires high runtime complexity, but it may be unnecessary to derive an optimal solution in many scenarios. Instead, approximate solutions, polynomial in computation complexity, are sufficient for most applications [58]. The results from approximate solutions may be sub-optimal, but they require less computation time because they drop results that consume the same amount of network resources but make a smaller contribution to the final solution. There are several methods that can derive approximate solution in polynomial time [19] or even constant time [18]. Some general forms and approximate algorithms are discussed in [59, 60].

The traffic distribution solution presented in this dissertation, *i.e.*, via joint routing and channel assignment, falls into this category of optimal routing algorithms. It targets the limitations of optimization-based routing solutions. The key contribution of this dissertation is to provide an adaptive traffic distribution solution that handles the dynamic traffic demands with fast approximation algorithms of low complexities that are suitable for distributed implementation.

Wireless Network Capacity

Wireless network capacity, or how much traffic can be delivered through a wireless network, is an important metric for the wireless network. It reflects the upper bound for the network routing performance. The wireless network capacity problem has been studied in several works [14, 38, 61, 62, 63, 64]. [14] studies network capacity in a wireless network with n identical randomly distributed nodes, and it found that the throughput obtainable by each node for a randomly chosen destination is $\Theta(\frac{w}{\sqrt{n \log n}})$ where w is the transmitted rate. [14] models a network using both *physical model* and *protocol model* to compute the capacity. It also analyzes the capacity of an arbitrary network. However, it considers single channel networks or multi-channel networks with only one radio at each node. In [38] and [61], the authors extend wireless network capacity analysis to multi-channel and multi-radio networks. [65] also studies the impact of the number of channels and interfaces to multi-channel and multi-radio networks and shows how wireless network capacity scales as the number of nodes increases. The existing literature also investigates other factors of network capacity. For instance, [62] studies the capacity of multi-channel networks with channel switch constraints, such as hardware complexity and spectrum use. Optimization methods in expanding WMNs with new nodes are discussed in [63]. At the same time, [64] presents a throughput capacity analysis of a specific flow where node location and interference are considered.

The capacities discussed above are the theoretical upper bound of the traffic that can be routed through a network. It depends on routing algorithms to provide a feasible solution to route the traffic and determine whether the bound is achievable or not. In reality, the actual network performance may be less than the capacity due to various reasons. The goal of this dissertation is to design a routing algorithm that can route traffic that is close to that bound.

Wireless Network Traffic Analysis

Traffic demand is an important component in determining wireless network routing algorithm performance. A good understanding of network traffic, which includes traffic patterns and user behaviors, can provide correct input for routing algorithms to compute optimal routing paths. Network traffic can be analyzed by studying its trace files, which contain network protocols, and flow information. By learning from these information sources, we can understand current traffic status and possible future traffic. [66] and [67] use network trace to analyze campus network traffic. Their studies show that the activity and traffic are dynamic and vary over different time scales and different locations. Also, their studies show that neither inbound nor outbound traffic dominates the network traffic, and the ratio also varies from time to time. Thus, asymmetric bandwidth design is not practical in their traces. Based on the analysis of trace files, it is possible to derive a formal model for network traffic. The work of [20] uses the Weibull regression model to characterize flow arrival and explains the implication from user perspective and application demands.

Network traffic is generated by users, and user behavior is a key factor that affects wireless traffic patterns. Studying user behavior, including user arrival/departure and user traffic demand, can help better explain the root cause of wireless traffic problems. [68] studies the user activities in a public wireless network and shows user arrivals are correlated in time and space. It also shows that most wireless users have short session time, for which the distribution follows the General Pareto distribution and medium bandwidth consumption is between 15 and 80 $Kbps$. [69] proposes a spatial model for mobile user registration patterns as they move from one access point to another within the same network. The study finds that user registration patterns show a distinct hierarchy, and access points can be clustered based on the user transition probability. Unlike in mobile wireless networks, mobility is rarely discussed in WMNs because WMN routing focuses on traffic aggregated at fixed access points, and most users limit movement to the area within a single access point's coverage [66, 67].

A good understanding of traffic characteristics can help to develop good traffic prediction algorithms, which can, in turn, provide accurate traffic information for routing algorithms. It should reliably predict general long term trends and be agile enough to catch dynamics at short terms. One popular mathematical tool for traffic modeling is time series analysis. This tool provides solutions for studying correlation in data by identifying the trend of data and the distribution of variation. When predicting data via time series analysis, one must consider the relationship with historical data as well as the overall trend and other variations. There are several factors that can facilitate traffic prediction, including historical traffic and network properties. Traffic analysis that is solely based on the data and time series models may not be accurate and efficient. Embedding the network context into the modeling may provide more insightful information about the traffic, help characterize network traffic (including correlation and seasonality), and ultimately produce more accurate traffic prediction. A detailed introduction of traffic prediction and time series analysis can be found in Chapter IV.

Optimization Frameworks with Uncertain Inputs

To develop a routing optimization solution that can handle uncertain and dynamic traffic demands, we need a mathematical optimization technique that can take uncertain inputs. There are two mathematical programming techniques that accomplish this: *Stochastic Optimization* [70] and *Robust Optimization* [71, 72]. The major difference between these two frameworks is their optimization objectives. In *Stochastic Optimization*, the objective is to find a solution that is optimized for the expected case. In *Robust Optimization*, on the other hand, the goal is to optimize for the worst case scenario. *Robust Optimization* is gaining in popularity for network routing research, and several extended works based on robust optimization have been proposed. For example, [73] proposes a two-stage robust routing algorithm for network flows that allows one to control the conservatism of solutions. A distributed version of robust optimization is proposed in [74] and robust discrete optimization is described in [75]. Compared to *Robust Optimization*, the application of *Stochastic Optimization* is less common in network routing due to the following reasons [75]: 1) a good estimation of data distribution is required and 2) the size of the problem grows quickly as a function of the number of scenarios.

The work by Wellons et al. [76] is based on a robust optimization framework and focuses on the worst-case performance guarantee. This algorithm provides robust routing performance in highly dynamic network environments where traffic demand is unknown and traffic behavior is hard to predict. However, its average network performance has a large gap to the optimal value.

On the other hand, our uncertainty-aware traffic distribution framework is closely related to *Stochastic Optimization*. We address the challenges of stochastic optimization using 1) a traffic prediction algorithm which provides accurate traffic demand estimation and 2) a fast approximation algorithm which provides a feasible solution with a low computational cost.

CHAPTER III

MODELS

This chapter introduces features and models used in WMNs.

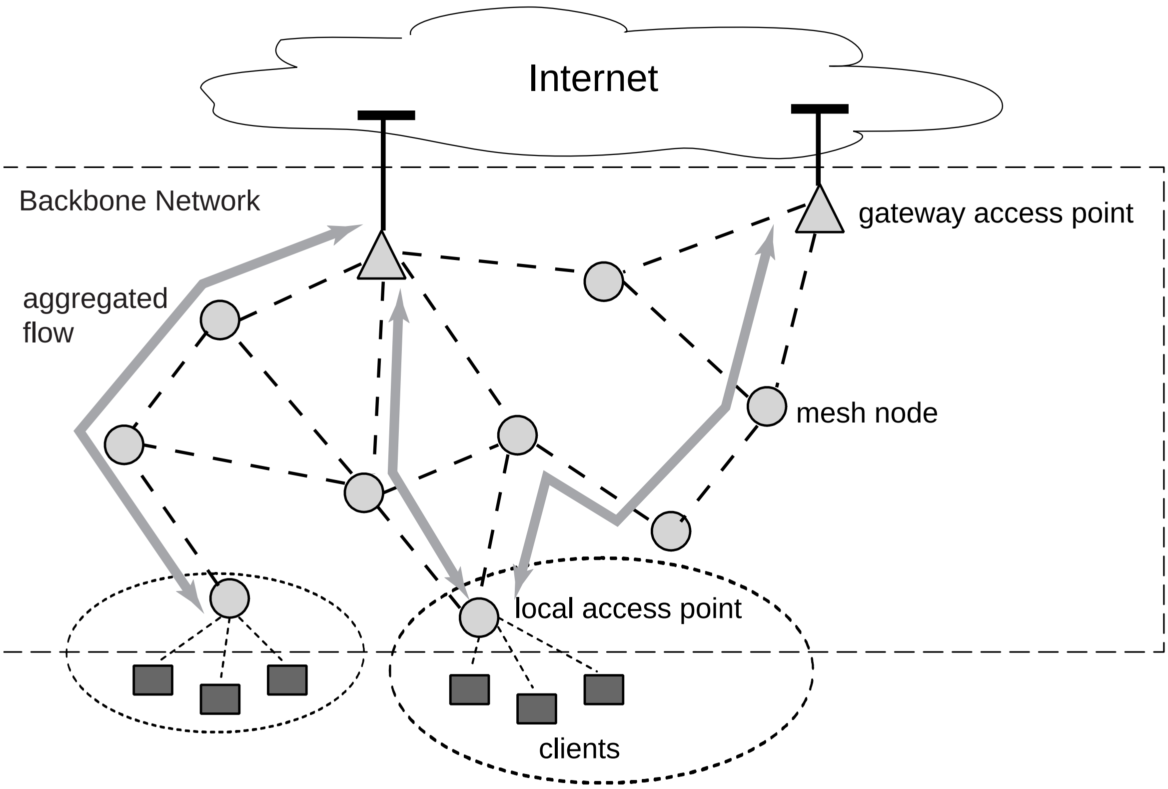


Figure III.1: Illustration of Wireless Mesh Network

Network Model

A multi-hop wireless mesh network is illustrated as shown in Fig. III.1. In this network, mobile devices, which are not the backbone of WMN, are connected to the WMN via local access points. Local access points aggregate the traffic from associated mobile clients. They communicate with each other and with stationary wireless routers, forming a multi-hop wireless backbone network which forwards the user traffic to gateway access points. These access points usually have wired connections to the Internet which send incoming traffic back in the reverse direction. In the following discussion, local access points, gateway access points, and mesh routers are collectively called mesh nodes. With the help of the WMN, mobile clients can access the remote Internet seamlessly.

For the sake of simplicity, we use the following notations: The backbone of a WMN can be modeled as a directed graph $G = (V, E)$, where each node $u \in V$ represents a mesh node. Among these nodes, $g \in V$ is one of the gateway access points that connect to the Internet.

Interference Model

Signal interference plays an important role in wireless network transmission. When one node sends out packets, its signal may be heard not only by the receiver, but also by some of neighboring nodes. When two nodes are far enough away from each other, the signal from each node cannot be heard by the other and no direct transmission is possible between these two nodes. At the same time, there is no interference between those two nodes, which means one node's communication will not affect the other. On the other hand, when two nodes are close enough, they can hear the signal clearly and communicate directly with each other. There is also an intermediate state, when two nodes may not be able to receive each other's signal clearly, which means direct communication is not possible, but the signal can still be detected; this is usually called noise. The noise received at one node can affect its transmission with other nodes. We say node A interferes with node

B when node B's receiving capability is disturbed by node A's signal. Sometimes, the noise at the receiving node is so strong that it prevents the node from receiving packets from others.

Several models have been proposed to describe this interference character. Two popular models, *protocol model* and *physical model*, define the condition for a successful wireless transmission and describe when two nodes are encountering interference [14]. The physical model is based on real scenarios and is much closer to actual networks. The protocol model, on the other hand, is a simplified, though popular, model favored in the research area.

Physical Model: When a node is transmitting packets to another node, its signal can be heard by neighboring nodes. The strength of the signal received at each node depends on the distance to transmitting nodes. Usually the farther it is away from transmitting nodes, the weaker the signal it receives. In the physical model, packet transmission from node u to v is successful if and only if: $SNR_{uv} \geq SNR_{thresh}$, where SNR_{uv} is the signal-to-noise ratio received at node v and SNR_{thresh} is the threshold value. Noise at node v consists of ambient noise and interference of transmission from other nodes in the network.

The physical model's description of signal interference is close to that in actual networks, where it completely depends on the signal-to-noise ratio at the wireless card to determine whether a node can hear the signal clearly and decode it. One node's signal may interfere with another node at one moment, but it may not interfere with that node at another moment. In the physical model, only the signal-to-noise ratio at the receiving node at a specific moment can determine whether it is disturbed by other nodes and whether it can receive the signal successfully. Although it models how interference is interpreted at the receiving node, the physical model is complicated and not easy to analyze. The protocol model, on the other hand, provides an easier modeling.

Protocol Model: In the protocol model, each node $u \in V$ has a transmission range (denoted by $R_T(u)$) and an interference range (denoted by $R_I(u)$). $R_T(u)$ is defined as the maximum distance node u's transmission can be received successfully by other nodes. Most models, especially those used in centralized algorithms, assume all mesh nodes have a uniform transmission range, and thus $R_T(u)$ can be simplified to R_T. We denote $r(u, v)$ as the distance between u and v. A directed edge $e = (u, v) \in E$ denotes that u can transmit to v directly. An edge $e = (u, v) \in E$ exists if and only if $r(u, v) \leq R_T$. If node u is transmitting packets to node v, u's signal can also be heard by node t if $r(u, t)$ is within a certain range, which is denoted by $R_I(u)$. The signal from node u is treated as noise by node t if t is not the intended receiver and the noise also successfully prevents t from receiving packets from other nodes. Most existing works that use this model also assume that all mesh nodes have a uniform interference range R_I. The relationship between R_T and R_I can be expressed as $R_I = (1+\Delta)R_T$, where $\Delta \geq 0$ is a constant.

Packet transmission from node u to node v is successful if and only if

1. the distance between these two nodes $r(u, v)$ satisfies $r(u, v) < R_T$, and

2. any other node $w \in V$ within the interference range of the receiving node v, *i.e.*, $r(w, v) \leq R_I$, is not transmitting.

Fig.III.2 shows an example in which nodes w, x and v are within the transmission range of node u. Transmission between any node from w, v, x and node u is valid. At the same time, nodes w, v, x, b and c are all within the interference range of node u. Although node u can only transmit information to node w, v, x, its transmission signal can also be heard by nodes from b and c. Each node within

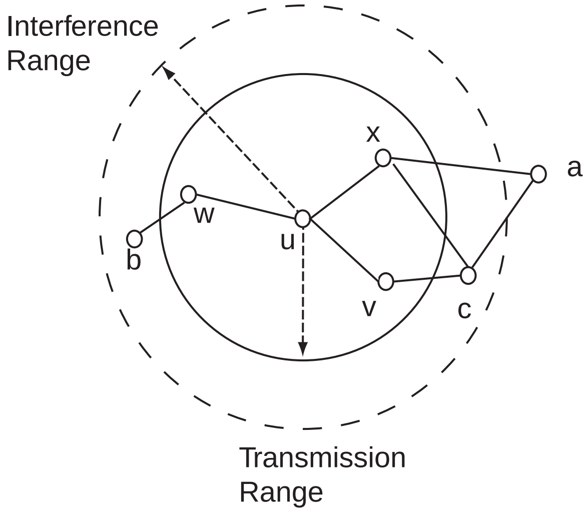


Figure III.2: Illustration of Transmission and Interference Range (node u)

u's interference range is disturbed when node u is in transmission and it is not the intended receiver. Node a is out of node u's interference range $R_I(u)$, and is not affected by node u.

Transmission always happens between two nodes, and the traffic can be in one direction or the other. When there is signal transmitting over a wireless link, it must be either one node or the other of the link sending the packet. Any external interference with any of the two nodes on the link also interferes with the transmission of that link. This means that when one link is in transmission it may affect the transmission of other links. All affected links are included in the *interference set* of that link.

When one node pair is in transmission, other nodes within the transmission range may not be able to transmit. Two edges e, e' interfere with each other if they cannot transmit simultaneously. We use $I(e)$ to denote the set of edges which interfere with edge e. Fig. III.3 shows an example of interference set $I(e_{uv})$. The circles show the interference range of node u and v. Edges e_{ab}, e_{xy} are all in the interference set $I(e_{uv})$ and encounter interference when edge e_{uv} is in transmission.

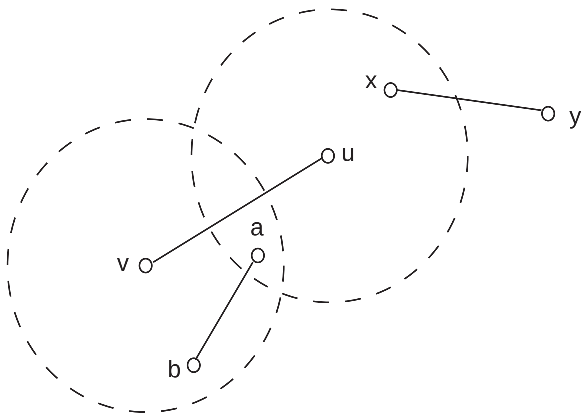


Figure III.3: Interference Set of Link vu

Multi-radio and Channel Assignment

Wireless interference has a great impact on throughput performance of WMNs. Simultaneous transmission is possible only when two links do not interfere with each other. The impact is amplified, especially in a multi-hop network where interference is heavy, when the number of transmitting links increases.

The IEEE 802.11 protocol provides several non-overlapping channels to address the simultaneous transmission problem in wireless networks. For example, 802.11b and 802.11g have 3 and 12 orthogonal channels, respectively. Those channels use the same transmit protocol, but they transmit on different frequencies and do not interfere with each other. This greatly broadens WMN capacity and increases the throughput. As long as they use the same channel, two nodes are able to communicate with each other. However, transmission is impossible if they operate on different channels. Since only one channel is allowed on each wireless interface card (also called *radio*), each node can be equipped with multiple radios and operate them on different channels to increase transmission capacity with other nodes.

The purpose of channel assignment is to assign a channel to both radios on each pair of nodes so that two nodes can communicate on the same channel. The channel assignment problem is different from the graph-coloring problem in that standard graph-coloring cannot capture the constraints in channel assignment [37]. One challenge is that the number of radios on each mesh node is limited due to design issues and may be far fewer than available orthogonal channels. Learning how to assign channels to limited radios to reduce the interference while increasing WMN throughput has become one of the major topics in this area.

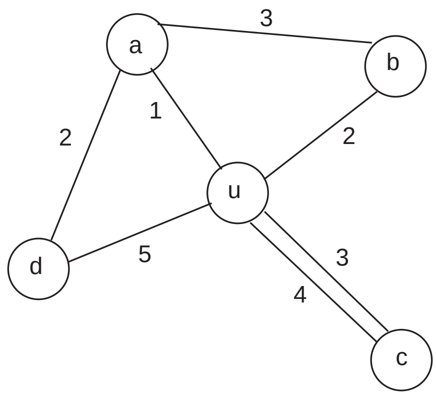


Figure III.4: Network Topology with Multi-radio

Fig. III.4 shows an example of network topology after channel assignment. There are five interfaces on node u. Three different channels are used to communicate with node a, b, d respectively. Two channels are assigned to node u for communicating with node c. It is valid to use more than one channel for single-pair node communication in order to increase the total transmission capacity. Channel assignment is not intended to prevent interference but to reduce the interference. As in this example, both link ad and bu use channel 2 for communication. It is still possible that those two channels may interfere with each other.

Schedulability

To study the optimal routing problem, we first need to understand the constraint of the flow rates. Let $\boldsymbol{y} = (y_e, e \in E)$ denote the wireless link rate vector, where y_e is the aggregated flow rate along wireless link e. Link rate vector $\boldsymbol{y}$ is said to be schedulable if there exists a stable schedule that ensures every packet transmission with a bounded delay. Essentially, the constraint of the flow rates is defined by the schedulable region of the link rate vector $\boldsymbol{y}$. For ease of exposition, we assume that the wireless link data rate, which is the maximum data that can be carried in a unit of time, is the same for all $e \in E$ and is denoted as q. q is also referred to as *channel capacity* in the following discussions.

The link rate schedulability problem has been studied in several existing works which have led to different models [77, 78, 79]. In this dissertation, we adopt the model in [78], which presents a sufficient condition under which a link scheduling algorithm is given to achieve stability with bounded and fast approximation of an ideal schedule. Based on this model, we define I'_e as a subset of I_e where each $e' \in I'_e$ has a length $r(e')$ greater than or equal to $r(e)$. We further define $S_e = \{e\} \cup I'_e$ as the *adjusted interference set* of e. Based on the scheduling algorithm and its properties presented in [78], we have the following claim.

Claim 1. (*Sufficient Condition of Schedulability*) The link rate vector $\boldsymbol{y}$ is schedulable *if* the following condition is satisfied:

$$\forall e \in E, \ \sum_{e' \in S_e} y_{e'} \leq q \qquad \text{(III.1)}$$

If we extend this model to multi-channel, multi-radio networks, we have the following claim.

Claim 1a. (*Sufficient Condition of Schedulability*) The link rate vector $\boldsymbol{y}$ is schedulable *if* the following condition is satisfied:

$$\forall e \in E, \sum_{e' \in S_e} y_{e'} \leq q \tag{III.2}$$

Some existing works also study the necessary condition for multi-radio, multi-channel networks. Formally, let $y_e(c)$ be the flow rate on edge $e(c) \in E_c$, $\boldsymbol{y}$ be the link flow vector, $\rho_e(c) = \frac{y_e(c)}{\phi_e(c)}$ be the utilization of channel c over link e, and $E(v)$ be the set of links that is adjacent to node v. Based on the results presented in [18], the necessary conditions of channel assignment and scheduling are summarized in the following claim.

Claim 2. (*Necessary Condition of Channel Assignment and Schedulability*) For the multi-channel, multi-radio wireless mesh network, if a given link flow vector $\boldsymbol{y}$ does not satisfy the following inequalities:

$$\sum_{e' \in I_e(c)} \rho_{e'}(c) \leq \gamma(\Delta); \forall e(c) \in E_c \tag{III.3}$$

$$\sum_{c \in C} \sum_{e(c) \in E(v)} \rho_e(c) \leq \kappa(v); \forall v \in V \tag{III.4}$$

then $\boldsymbol{y}$ is not schedulable.

In particular, Inequality (III.3) is the *congestion constraint* over an individual channel. $\gamma(\Delta)$ is a constant that only depends on the interference model. Inequality (III.4) gives the *node radio constraint.* Recall that a mesh node $v \in V$ has $\kappa(v)$ radios, and thus can only support $\kappa(v)$ simultaneous communications.

Wireless Link Quality

Wireless networks are different from traditional wired networks in several aspects, including wireless interference and link stability. It has become important to study those behaviors and properties in actual wireless networks before designing routing algorithms. Questions such as how much traffic can be routed through a network or a specific link are critical to overall system performance. Several research endeavors (*e.g.*, [57, 80, 81, 82]) have focused on studying wireless link properties in deployed wireless networks.

Unlike wired links that are relatively stable during transmission, wireless links have an instability problem. Transmission between two nodes may succeed at one time but may fail at another time for various reasons. One metric measuring link transmission quality is the link loss rate, which is the rate of packets not received by the receiver. The link loss rate is an important factor that reflects the link quality, including the number of retransmissions and the actual link capacity. It [57] has been shown that link loss rates in a wireless network are usually uniformly distributed, and there is no absolute threshold distinguishing whether two nodes are "in range" or "out of range." According to their results, the protocol model, a simplified model introduced earlier, does not reflect the actual link connectability.

There are several factors that can affect the link loss rate. Signal-to-noise ratio, which reflects the signal interference, and distance, which reflects the signal attenuation, are both important factors that make major contributions to the link loss rate. Some studies [57] also show that multiple-path fading is a determining factor for the links that have intermediate loss rate. Other studies [83] shows that wireless interference and contention may even starve links with nodes at certain locations.

Models [82, 81] based on the measurement of deployed networks have been proposed to study link quality. [84] presents the link capacity of any given link in the presence of any given number of interferences in a deployed network. In [82], models based on measurements of wireless network signal characteristics are used to predict the performance of networks with different settings. On the other hand, models may not be able to reflect actual link conditions. [81] shows that inaccurate signal characterization will result in poor network performance. The loss rate of a specific link is not constant and may change over time. This characteristic is supported by research in [80], which shows increased variability in channels at a time scale that is smaller than a single packet increases the link-level throughput while longer time scale variability reduces it.

CHAPTER IV

WIRELESS NETWORK TRAFFIC ANALYSIS

Many existing routing algorithms for WMNs that take traffic information as input assume it is static or known a priori. However, studies (*e.g.*, [20, 85, 86, 87, 88, 68, 89, 90, 91]) show that traffic demand, even aggregated at access points, is highly dynamic and hard to estimate.

Such observations reveal the limits of the algorithms that are not adaptive to dynamic traffic because those algorithms may not only generate sub-optimal solutions, but they can also produce even worse performance due to the mismatch between the demand used in routing and the actual traffic. Consequently, it has become more and more important to provide precise future traffic estimation so that routing algorithms can compute optimal routing paths. A straightforward approach for traffic prediction is to predict future traffic by observing historical data. A simple example for this approach is using traffic at a previous time slot as the prediction for the current slot. More often, traffic cannot be predicted based on previous time slot, and a more general formal approach is needed.

In order to study the relationship between historical and future data, a common metric called *autocorrelation* is used. This metric reveals how strong the correlation is between sub-series at different time points from the same data series. Using this concept, we can develop models based on observed traffic information and then predict the traffic based on the developed model. Time series analysis is one such methodology. Time series analysis is a tool for studying, modeling, and predicting a series of data. It uses stochastic approaches to model historical data and predict future traffic, and it has been extensively studied in many references and books [92, 93, 94, 95, 96, 97, 98].

Traffic estimation can predict future traffic demand by studying historical data as well as the network environment. Traffic prediction models can be built either based on specific system property, or historical traffic demand, or both. In [68, 87, 20, 89], traffic predictions are made by studying user behaviors or traffic patterns at the flow level. As another approach, traffic can be analyzed through decomposition because the variation at different timescales are usually caused by different network mechanisms. In [99], traffic is decomposed into different scales and each decomposed traffic is predicted independently using time series analysis models. In [20], two-tier modeling based on different time granularity is used to capture the nonstationarity characteristic of wireless network flows. *Weibull Regression* model is then applied to both time scales to characterize the traffic. The benefit of using system properties is to provide traffic prediction based on internal factors that cause the traffic change, which can usually produce precise traffic prediction, especially for patterns that change rapidly. However, system properties may vary from one to another, and it would be difficult to develop a general solution for all different situations.

In most scenarios, traffic prediction provides a single determined value for future traffic at a specific time slot based on the historical data. However, this can be insufficient since the prediction result is probabilistic in nature. In order to address this problem, we present a novel method to describe the future traffic. Instead of relying on a single value to predict future traffic, we use a set of possible traffic values together with corresponding probabilities.

In this chapter, we first present the wireless network trace files, which we will use for our routing algorithms for simulation and verification. Then, we introduce the concepts of data correlation, time series analysis method, and other approaches that can be used to analyze and characterize network traffic. Using these tools, we then present the mean value traffic prediction method, which generates single-value traffic prediction, and mean value traffic prediction with statistical distribution method, which provides corresponding probabilities for predicted traffic demand values.

Trace Data Sets

In order to develop realistic traffic models, it is very important to have traffic information that is close to that of real networks. One of the best approaches is to use the trace files from real networks. Crawdad [100] is a popular website that maintains wireless network traces collected by different institutions and groups. Those traces are generated by different networks and environments, which include wireless LANs, ad-hoc networks, and WMNs. Those networks include large-scale systems that are deployed in colleges and public locations, as well small-scale systems, such as homes. The length of the trace files varies from several hours to several years, depending on different properties of networks. Traces from temporarily deployed networks, such as a Wi-Fi network for a conference, have relatively short length, while permanently deployed networks may collect longer traces. The contents of the trace files include traffic load, location information and signal strength. The trace file formats also vary depending on the purpose and content of each file and the way the trace file is collected. Common file formats include tcpdump, SNMP, and Syslog, which are usually produced by common tools like sniffer software tcpdump. Some traces generated by self-developed programs may use proprietary formats to record trace files.

Our study focuses on the traffic routing in the backbone WMNs, and it requires that traffic load information be collected at access points as the traffic demands. The preferred trace data that match our needs are traffic collected from WMNs. However, only limited traces from WMNs are available on Crawdad. On the other hand, several traces collected from wireless LANs are available. We believe traffic traces from wireless LANs also qualify our study for the following reasons. First, we are only interested in traffic demands that are gathered at access points. The information about how traffic is routed through wireless routers is not important from trace files. Second, from an end users' view, the network structures of wireless LANs and WMNs are similar to each other. Also, users in both networks have similar behaviors. Therefore, it is reasonable to use wireless LAN traffic for our study.

We are primarily interested in SNMP, tcpdump file formats, which keep the network traffic load information. We prefer trace files in SNMP format because it gives detailed and complete traffic information at each node and also provides an easier way to extract traffic information from the trace file. Trace files, like those from tcpdump, do not track the traffic flow directly through a specific access point directly, so extra work is required to extract the traffic information.

Based on the criteria described above, we summarize the data sets that are used in our study below.

dartmouth/campus This data set contains complete wireless network information from Dartmouth College for several years. This is the most complete data set on Crawdad with the longest duration. The formats of the trace files include syslog, SNMP, and tcpdump. However, trace files with all three formats are only available for the early years' (2002 - 2003) traffic. A problem with early years' wireless traffic is that wireless technology was not as popular it is now, and traffic shows more random trends due to a low number of users. Only syslog files, which do not have detailed traffic information, are available for recent years, and cannot be used for our study.

ibm/watson It is a wireless LAN data set from the IBM research center recorded in 2002. The duration for the trace file is several weeks and the format of the trace files is SNMP.

Mean Traffic Prediction

With the help of trace data from real networks, it is possible for us to study traffic demand characteristics and develop traffic prediction algorithms. Understanding traffic is the first step for a traffic prediction algorithm, which gives us general pictures of the data sets and how data are linked.

Understanding Traffic

Preliminaries

As a preliminary step of finding traffic patterns, it is necessary to study its correlation. In statistics, correlation is defined as the relationship between two or more sets of variables, and autocorrelation of a traffic series within this dissertation is the correlation between its subseries starting at different times. There are several ways to study traffic autocorrelations. One basic approach is to assume one traffic series is a function of the other with a different time offset and tries to find a model that fits two traffic series relationship.

Fig. IV.1 illustrates an example of finding traffic correlation between traffic at current time and previous hour. The x axis represents the traffic at current hour and the y axis represents the traffic at previous hour. It is easy to see that the

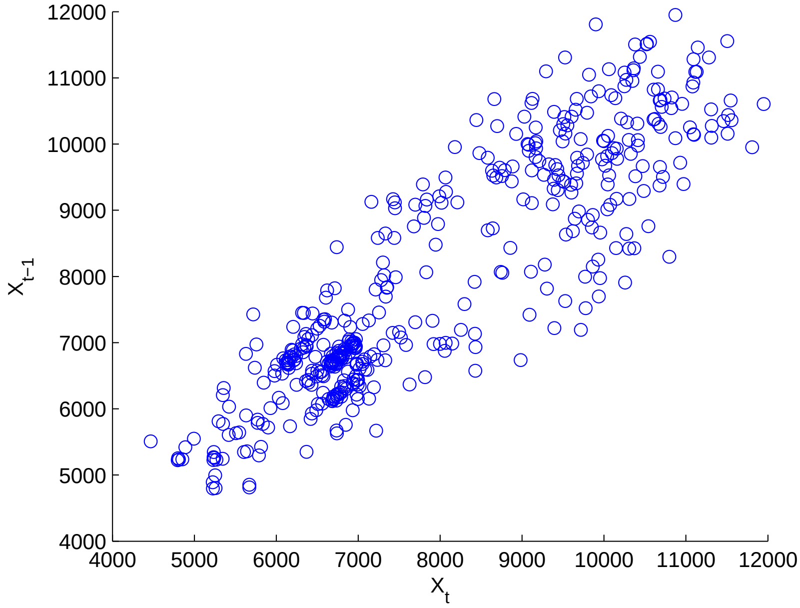


Figure IV.1: A Basic Approach for Studying Traffic Correlation

data plot shown in the figure indicates a strong linear relationship, and we can use tools such as least squares to find the linear relationship that best fits data and show how strong the linear relation of the traffic is with its previous hour.

Correlation

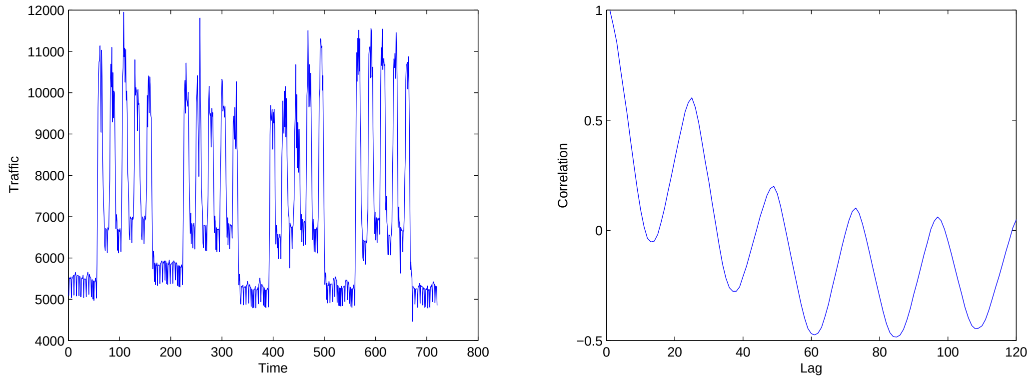
The above approach provides a basic tool for studying the relationship within a traffic series. However, the weakness of this approach is that it can only study relationships between a limited number of sub-series at each time. It also depends on individual experience to choose which sub-series may have strong relationships and analyze them. In statistics, autocorrelation describes how sub-series with different time lags are similar to each other. Let $\mu = E(X)$ be the mean of a time series X, and σ be its standard deviation. We define the autocorrelation ρ_τ of two sub series X_t and $X_{t+\tau}$, starting at time point t and $t + \tau$ respectively, as

$$\rho_\tau = \frac{E[(X_t - \mu)(X_{t+\tau} - \mu)]}{\sigma^2} \qquad \text{(IV.1)}$$

$E[(X_t - \mu)(X_{t+\tau} - \mu)]$ is also known as autocovariance and τ is the lag. A higher autocorrelation value indicates a stronger correlation. It is easy to see that the autocorrelation of a time series X_t itself is 1 when τ is 0, which is the strongest correlation.

The above definition assumes continuous traffic series, and we also assume that the traffic length (duration) is large enough that the difference caused by τ can be ignored. However, in our data analysis, which is obtained from network traces, the collected traffic series is discrete, and its length is finite. For a discrete traffic series with length N, we refine the previous definition of X_t and $X_{t+\tau}$ as sets of $\{x_t, x_{t+1}, ..., x_{N-\tau}\}$ and $\{x_{t+\tau}, x_{t+\tau+1}, ..., x_N\}$, respectively, where x_t is the traffic at time point t so that two sub-series share the same length. All other variables in Equation IV.1 remain unchanged and use values from the original series.

Beyond Correlation



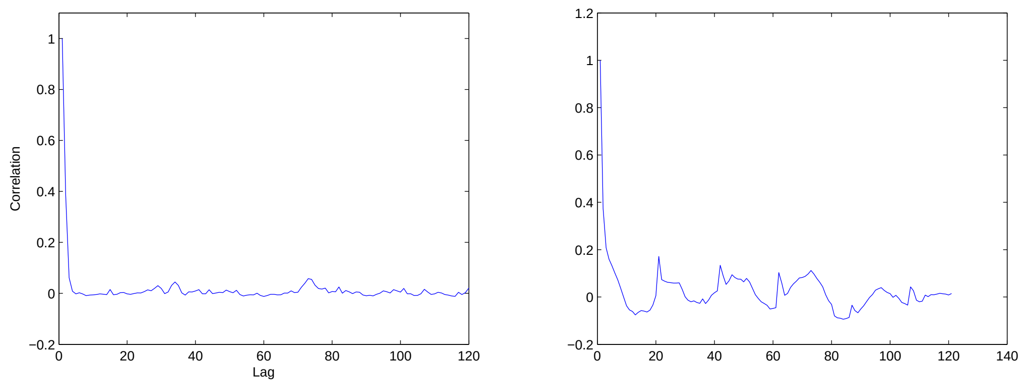
(a) Original traffic series (b) Autocorrelation

Figure IV.2: A Traffic Series and its Autocorrelation

Fig. IV.2 is an example of a traffic series. Fig. IV.2 (a) is the original traffic and Fig. IV.2 (b) is its autocorrelation. The traffic is recorded hourly, and the lag used in the x axis is also indexed by hour. It is obvious to observe from the figure that there are peaks around hour $24, 48, ..., 120$, which means traffic at time t has stronger correlations with traffic at $t + 24, t + 48, ..., t + 120$. This result further indicates that traffic at the current hour has stronger correlation with traffic at the same hour of previous days. At the same time, the values at lag $1, 2$ are high, which leads to the conclusion that the traffic at previous hours also has stronger correlations with the traffic at the current hour.

The above example is from an access point trace in a typical corporation network, which renders certain patterns in terms of correlation. The data series itself determines its correlation; however, the way we collect data from network traffic trace can generate different series and thus lead to different correlation results. For example, access points at different locations may show different traffic patterns and correlations. Series from corporation networks may have different correlations than those from campus networks. Also, different time scales used for measuring traffic can lead to different autocorrelation results. Traffic aggregated by minutes may show no strong correlation among different minutes, but when aggregated by hours, it may show stronger autocorrelation.

We collect traces from various network environments, and those traffic series can vary from one to another. Even in the same environment, different Access Points (APs) may have different traffic behaviors. Fig. IV.3 illustrates an example where two APs from the same network show different autocorrelations. Series A shows almost no correlation among historical hours. It is more like a random traffic pattern, and it cannot be analyzed using an autocorrelation approach. Series B, however, shows a better correlation among historical hours than series A. Since



(a) Traffic Series A (b) Traffic Series B

Figure IV.3: Autocorrelations from Different Traffic Series

different traffic series have different traffic patterns, it is impractical to derive a single traffic pattern model for all APs. Analysis based on individual traffic sources is necessary.

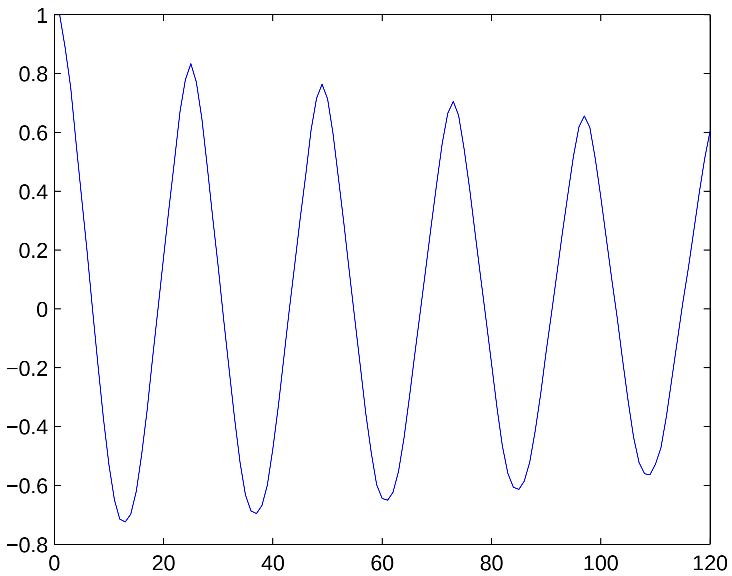
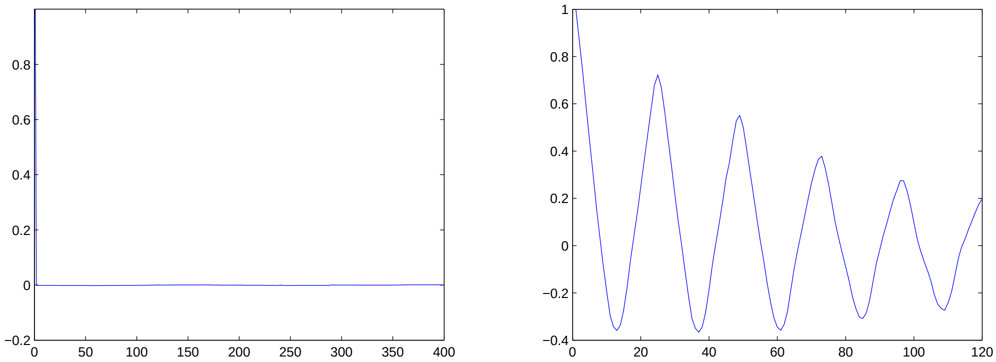


Figure IV.4: Traffic with Weekend Traffic Removed

Different traffic series have different patterns, and data inside a traffic series can also have different behaviors. It is difficult to study those series if we use the complete series as a single one. Decomposing those data based on different traffic behaviors, grouping data that share the same characteristic, and studying those series group by group can provide a better picture of traffic and make analysis more efficient. Fig. IV.2 is a typical traffic trace from a company. From our previous analysis, we know traffic have certain correlations with those from previous days at the same hour, but the correlation is not strong. In that figure, we use the whole set of traffic data, including weekday and weekend traffic. However, we know the employers of most companies do not work over the weekends, so weekend traffic can vary greatly from weekday traffic. By removing weekend traffic from the trace, we keep all weekday traffic, which has closer patterns than weekend traffic. Fig. IV.4 is the autocorrelation diagram after weekend traffic is removed. It shows a stronger correlation with traffic from previous days at the same hour.

Besides different patterns, it is also critical to choose appropriate time scales to measure traffic. Using larger or smaller time scales that are not compatible with actual system properties may hide or remove traffic correlations and make it impossible to study traffic relationships. Fig. IV.5 is an example of how important it is to choose a proper time scale to measure traffic. The original trace file records traffic every 5 minutes. If we use this traffic information directly and compute its correlation, we get the result as shown in Fig. IV.5 (a). It is easy to observe from the figure that little correlation exists if traffic is recorded every 5 minutes. However, we can obtain a better correlation result as shown in Fig. IV.5 (b) after we aggregate traffic hour by hour. This indicates that the time scale based on every 5 minutes is not a good metric for this trace file. Different traces represent different system properties, and a proper time scale can be estimated by studying the system properties and the trace files.



(a) Autocorrelation based on every 5-min traffic (b) Autocorrelation based on hourly traffic

Figure IV.5: Autocorrelations Using Different Time Scales

Traffic Prediction

With this understanding of data series patterns, we now apply mathematical tools to model the pattern and use the model to predict future traffic. As we introduced in the previous section, one of the tools is regression. By setting up a sub-series as a function of others, regression tries to find a fit function that can best describe the relationships of those series. Regression, however, can only handle simple data that a single function can fit directly. This is complicated by the fact that network traffic is not simple and contains complicated relationships. Time series analysis is a tool that addresses the problem where data contain more than one level of correlation and patterns. It processes series by decomposing data into different components and modeling each component. Then it uses the models to predict the future traffic.

Regression

Regression uses one sub-series as a dependent variable and one or more sub-series as independent variables, and then it tries to find a fit function that satisfies those variables and can best describe the relationship between them. Common fit functions include linear and polynomial functions; thus, corresponding linear regression and non-linear regression compute the best fit functions. Besides those regressions, there are also other regression techniques addressing certain specific problems, such as robust regression where abnormal data are dropped during the computation.

We illustrate an example to show how we can use regression to predict future traffic. First, we believe there is certain correlation between the current hour traffic and its previous hour in our example series. In order to study the relationship, one can use the series of current hour traffic as the dependent variable and the series of corresponding previous hour traffic as the independent variable. Fig. IV.6 plots the data of those two series where x axis is series x_{t-1} while y axis is series x_t. Based on the observation of the plot, we can see these two sub-series have linear relationship, and the red line plotted in the figure is the result computed from linear regression. Based on this result, we can use the linear model to compute next hour traffic using current hour value.

From the example given above, we know that regression is a simple tool to model series relationships, and that it uses the model to predict future traffic. There are also problems with using this approach. The first problem comes from the fact that users need to determine which sub-series have strong correlations. Regression can only find the best fit function for given sub-series and show how close they are compared to that function, but it leaves users to choose the right

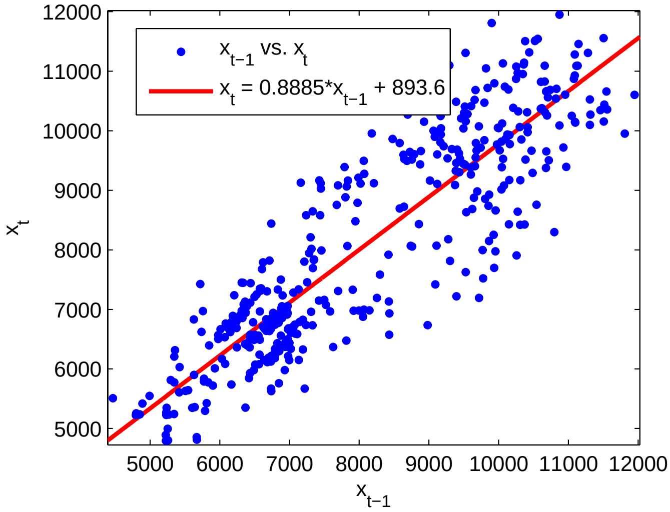


Figure IV.6: Traffic Regression Prediction

sub-series. Second, regression cannot process data with complex correlations or data patterns. For example, it is inaccurate to predict monthly sales data without removing some abnormal values caused by a holiday shopping burst. In order to address those issues, we introduce time series analysis for advanced data analysis in the next section.

Time Series Analysis

A time series is a collection of data observed over time. Typical examples of a time series include monthly airline passengers and yearly sale numbers. Time series can help find models for a better understanding of known data, and also use the model to predict future data.

A classic time series is usually composed of the four following components:

Trend: Trend in a time series is a component that has a steady growth or decline, and it reflects long-term changes in a time series.

Seasonal variation: Seasonal variation is a trend at a short time interval, and it repeats along the whole time series. A typical example for seasonal variation is a sales pattern of a product which may always be high during winter months and low in summer.

Cyclic variation: Cyclic variation is a trend at a time scale different from seasonal variation, and it may vary from cycle to cycle.

Irregular activity: Irregular activity is what is left after removing the trend, seasonal variation, and cyclic variation. It is usually a random component beyond forecast.

The value in a time series X_t at time t is a result of the above four components at that time point: trend T_t, seasonal variation S_t, cyclic variation C_t and irregular activity I_t. *Addictive model* and *multiplicative model* are two direct models for describing the combination of the four components. In *addictive model*, X_t is expressed as: $X_t = T_t + S_t + C_t + I_t$, while in *multiplicative model*, X_t is multiplication of the four components: $X_t = T_t \times S_t \times C_t \times I_t$. In some models, trend T_t and cyclic variation C_t are considered jointly as a single component as TC_t.

Time Series Models: Models for describing time series are simple when data is dominated by stationary trend and/or seasonality. However, those models are less efficient when the trend and/or seasonality are changing or successive irregular activity values are correlated. In those time series, successive values in the same series, usually in a short time interval, show certain correlation. The correlation within a series is generally called autocorrelation. More sophisticated models that are based on the relationship of successive data are needed.

A series is deterministic if values in the series can be determined completely based on past values. Most time series, however, are stochastic, which means they can only be determined partly by past values. However, it is still possible to calculate the probability of a future value. Such a process is called a stochastic process, and the model used is a stochastic model. An important class of stochastic models is called stationary models, where the mean is constant through time. In reality, there are many time series that do not have constant means and are called nonstationary process. However, analysis for nonstationary models is usually specific to certain series and may not be general to all. More analysis is actually based on stationary models, and nonstationary models can always be transformed into corresponding stationary ones.

We next introduce some common models for describing time series.

Autoregressive (AR) Model AR model is a stochastic model, where the value X_t at time t is the weighted sum of past p values plus a random shock a_t. That is

$$X_t = \phi_1 X_{t-1} + \phi_2 X_{t-2} + ... + \phi_p X_{t-p} + a_t \tag{IV.2}$$

Then, the series X is an AR process of order p. The simplest AR model is when $p = 1$. Then AR(1) can be simplified as

$$X_t = \phi_1 X_{t-1} + a_t \tag{IV.3}$$

AR(1) is stationary when $|\phi_1| < 1$. From Equation IV.3, we find that the only data that will affect X_t at time t is its precedent X_{t-1}.

If we use a lag operator L to represent the relationship between two consecutive variables, $LX_t = X_{t-1}$, then Equation IV.2 can be rewritten as follows:

$$\phi(L) X_t = a_t \tag{IV.4}$$

where $\phi(L) = 1 - \phi_1 L - \phi_2 L^2 - ... - \phi_p L^p$

Moving Average (MA) Model MA model is a stochastic model, where the value X_t at time t is the weighted sum of last q random shocks. That is

$$X_t = a_t + \theta_1 a_{t-1} + \theta_2 a_{t-2} + ... + \theta_q a_{t-q} \tag{IV.5}$$

Then we call the series X an MA process of order q. If we define $\theta(L) = 1 + \theta_1 B + \theta_2 B^2 + ... + \theta_q B^q$, then Equation IV.5 can be rewritten as follows:

$$X_t = \theta(L) a_t \tag{IV.6}$$

Autoregressive Moving Average (ARMA) Model In order to achieve greater flexibility of fitting time series, a mixed autoregressive and moving average model called ARMA model is needed. An ARMA (p, q) model is a mixed model that has p autoregressive terms, and q moving average terms and it is usually expressed as

$$\phi(L) X_t = \theta(L) a_t \tag{IV.7}$$

where the definitions of $\phi(L)$ and $\theta(L)$ are the same as in Equation IV.4 and IV.6.

Autoregressive Integrated Moving Average (ARIMA) Model Many time series actually are not stationary, and AR, MA and ARMA cannot be applied directly to those series. One way to process those series is using differencing to convert *nonstationary* series to *stationary* ones. The converted series after the first differences at time t is $X_t - X_{t-1} = (1 - L)X_t$, where X_t is the value from original series. It is possible that a *nonstationary* series may need to apply differencing more than once to become stable. The series after the dth differences can be expressed as $(1 - L)^d X_t$. An ARIMA (p, d, q) model for a time series means the series can fit into an ARMA (p, q) model after applying differencing d times.

Seasonal Autoregressive Integrated Moving Average (SARIMA) Model

A seasonal ARIMA may apply if a time series has a repeated pattern with s time periods over the time. In the SARIMA model, L^s is an operator such that $L^s X_t = X_{t-s}$. A typical SARIMA $(p, d, q) \times (P, D, Q)_s$ model usually includes non-seasonal terms with order (p, d, q) and seasonal terms with order (P, D, Q).

Model Building

To build a model for a time series, the first step is to identify a proper one. Actual time series may never be stationary initially, and thus the ARMA model cannot be applied directly to the series. Those nonstationary time series usually contain trend and/or seasonality, and additional processing is required to remove trend and/or seasonality.

Seasonality of a time series can be found and verified by calculating its autocorrelation function and the actual mean of the series. For example, in a time series of monthly airline passenger numbers, a repeated peak season pattern can be found every month. It can also be verified by calculating its autocorrelation function; strong correlations can be found every 12 months. A reversed process can also be used to confirm the actual mean of a time series. After the seasonal pattern of a time series is found, it can be decomposed from the original series.

If a time series is still *nonstationary* after removing the seasonality, then there may be trends in the series. Using differencing to build a new series can remove the trend and make it *stationary*. Sometimes, it may take more than one differencing before the series becomes *stationary*.

The preprocessed series, after removing seasonality and trends, should be a *stationary* process, and can use the ARMA(p, q) model to fit the series. The first step of applying the ARMA(p, q) model is to identify the values of p and q, which is the same as identifying the AR(p) and MA(q) models, and it can be found by studying a series' autocorrelation function and partial autocorrelation function. In reality, the values p and q are usually small for actual time series. By studying all ARMA(p, q) models of possible combinations, unique features can be found in each type of model. Possible matching models can be found by comparing a series' autocorrelation function and the partial autocorrelation function with those unique features. A detailed description of the unique features of the autocorrelation function and partial autocorrelation function in each model can be found in [92]. After possible matching models are found, the next step is to calculate the parameters of the models to get a complete model for the series. Parameters can be computed based on best fitting theory such as minimum squares. When more than one model can fit a series, one approach is to pick the model that best fits the data.

As a last step in verifying that the identified model is correct, the computed model should be put back into the series to calculate its residuals from the original series. The residuals from a correct model should be independent and have no obvious correlation with each other.

In our traffic model, the strength of our traffic prediction algorithm also relates to how dynamic traffic is. If traffic is highly dynamic and there is no correlation between current and historical traffic, then the traffic prediction algorithm does not work. However, it does not mean that the traffic prediction algorithm is not adaptive to dynamic traffic environment. The sensitivity of traffic prediction algorithm relies on the window size of historical data as well as the weight of each historical data, both of which contribute to the final prediction. A larger weight of recent traffic implies a prediction algorithm that is more sensitive to the recent change. A larger weight of older traffic, on the other hand, is less sensitivity to recent traffic. Different traffic environments produce different traffic modeling for prediction. Those parameters from models determine the sensitiveness to traffic change.

Example

In this section, we give a general example of how to process data from trace files, model the data using time series analysis, and predict the traffic using the model.

Some trace files track networks for several days or several weeks, but the traffic information is usually recorded every several minutes or less. Such a fine granularity may not fit our study because traffic variation at minute level usually does not show a strong pattern. Aggregated traffic within each hour, on the other hand, can smooth the irregular traffic patterns at minute level and show a more consistent traffic trend. Also, using traffic information at minute level requires routing paths to change at the same frequency. Frequent routing table updates will introduce extra overhead to the system. Therefore, we believe measuring traffic at hour level best fits the system implementation. We use aggregated minute based traffic within the same hour to represent the traffic at that hour.

Fig. IV.7 shows traffic information collected at an AP for about 480 hours. Overall, the traffic has a repeated pattern every 24 hours. We use the first 240 hour trace (Fig. IV.8) as the history data to build a time series model. The length of the series for history data varies depending on the actual traffic. Various factors help determine the right time period for training. A shorter time period may be not enough for algorithms to learn the behavior, while a longer time period could introduce longer computation time complexity. We choose the best time period by identifying network traffic behavior so that history data is enough for traffic prediction algorithms to learn without imposing too much computational load. For online algorithms, where it is not possible to learn historical data at the beginning, time period can be adjusted as traffic prediction algorithms try to adapt dynamic traffic.

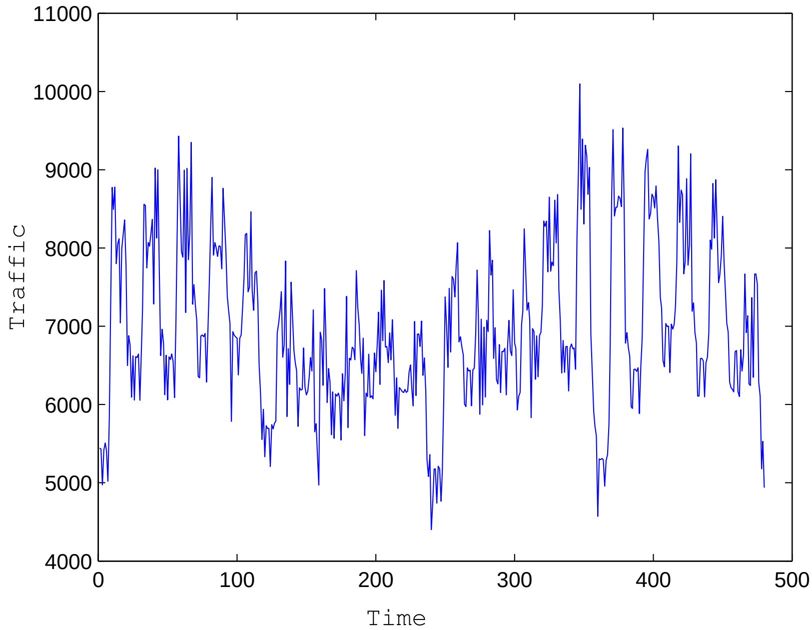


Figure IV.7: Hourly Traffic

According to the calculation of the trace and our understanding of the whole system, we find there is no obvious trend in this series, although the series shows seasonal variations. The first step is to find the seasonal trend of the series (Fig. IV.9), which can be calculated based on the same hour in each period (24 hours in this series), and remove it from the original series ((Fig. IV.10)).

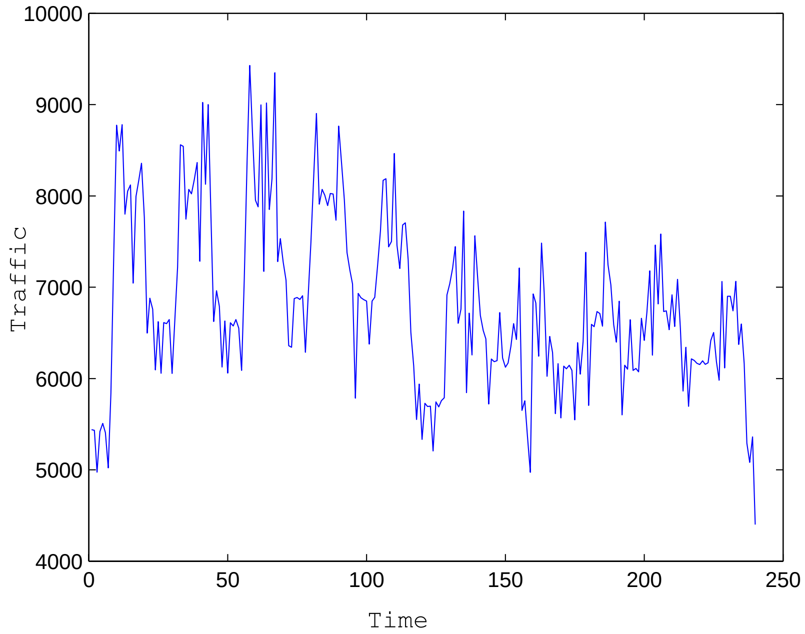


Figure IV.8: First Half Traffic

We use software programs (Matlab) to find the best model that fits this series, and the fitted model with seasonality included can be found in Fig. IV.11. The final fitted traffic with seasonality included is shown in Fig. IV.12.

Finally, we use the model developed from the first half of the trace to predict the second half (Fig. IV.13), and the overall traffic together with the fitted model and predicted traffic can be found in Fig. IV.14.

Mean Traffic Prediction With Statistical Distribution

Not all traffic can be predicted precisely, and this is particularly true when network traffic is highly dynamic and hard to predict. At the same time, incorrect traffic prediction provides wrong information for the routing algorithm and can affect network performance. In this case, single-value mean traffic prediction is not sufficient to address the problem. On the other hand, mean traffic predictions with

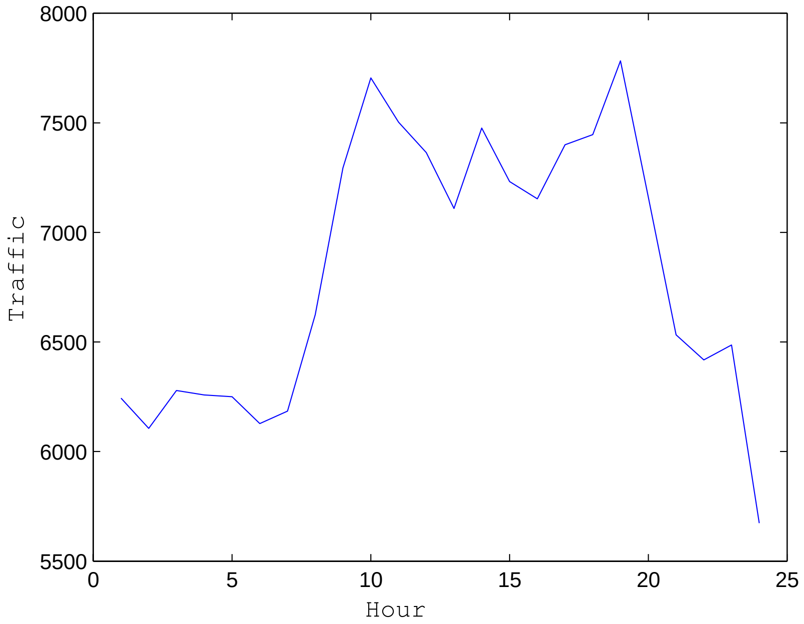


Figure IV.9: Average Seasonal Traffic Pattern

statistical distribution can properly capture this limit of mean value prediction. Instead of providing single value traffic information, it characterizes traffic by using a set of possible traffic values and corresponding probabilities. This approach is based on the mean traffic prediction. By comparing the deviation of predicted value from actual traffic, we can get a picture of error distribution in the traffic prediction. With this error distribution, we derive the distribution function of the predicted traffic demand. Traffic distribution algorithms that utilize the probabilistic traffic distribution information are more resilient to traffic dynamics.

A Complete Example

In this section, we describe a complete example from trace data analysis to traffic modeling to predict traffic with mean values and distribution. We study the dynamic behavior of aggregated traffic at local access points. Our goal is to

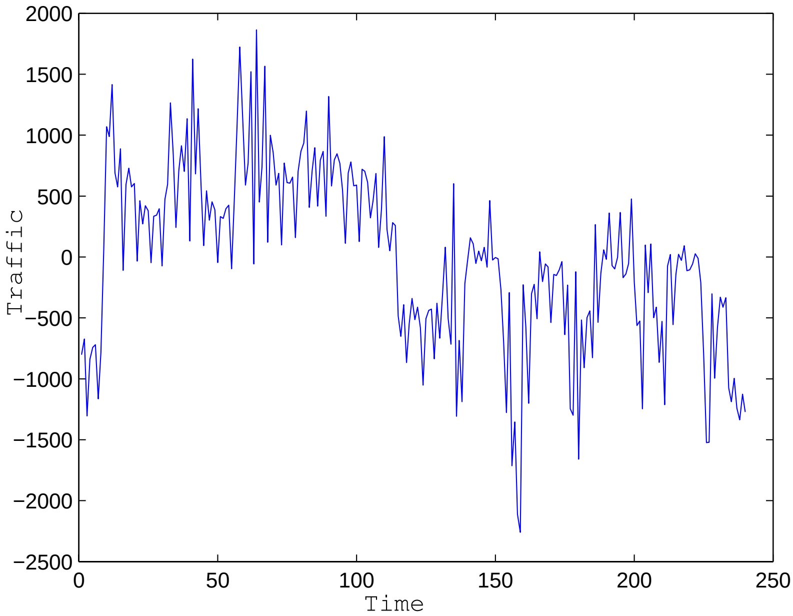


Figure IV.10: Traffic After Removing Seasonality

(1) develop a reliable estimation method that is able to predict the aggregated traffic demand of an access point based on its historical data, and (2) develop a statistical model to characterize the prediction results. The estimated traffic demand will serve as the input of mesh network routing algorithms which will be presented in the chapters to follow.

In order to develop such a traffic demand model, we study the traces collected at the campus wireless LAN network of Dartmouth College in Spring 2002 [100]. By analyzing the *snmp* log from each access point, we derive the dynamic behavior of the aggregated traffic demand. We argue that the access points of a wireless LAN serve a similar role and thus exhibit similar behavior because the local access points of a wireless mesh network as both networks server similar mobile clients.

To illustrate our analysis procedure, we choose one of the access points (ResBldg97AP3) as an example. The time series of its incoming traffic is plotted in

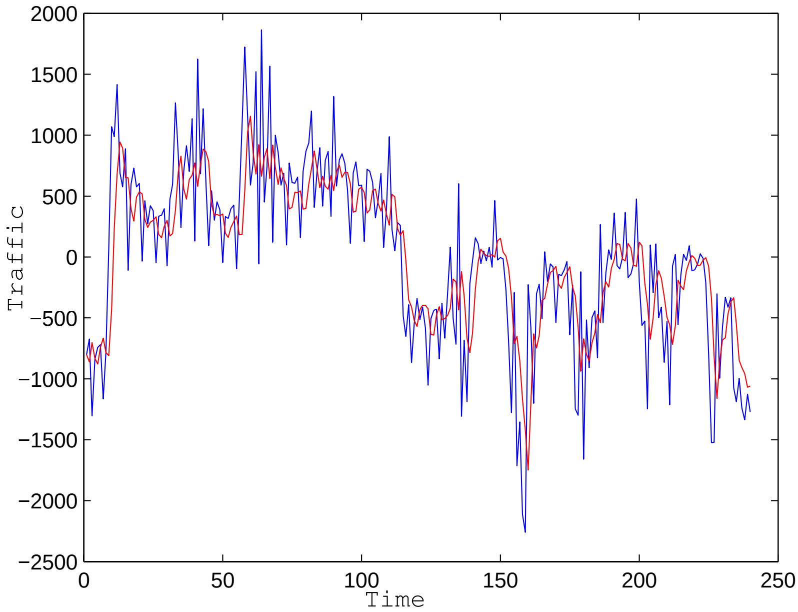


Figure IV.11: Raw Prediction Based Traffic without Seasonality

Fig. IV.15. From the figure, we can easily observe that (1) the traffic demand is non-stationary over large time scales due to the diurnal and weekly working cycles; (2) compared with the traffic behavior in the backbone Internet [101], the traffic at an access point is bursty due to the insufficient level of multiplexing. The above observations are consistent with the findings in [20].

The first step of our analysis is to identify and remove the daily and weekly cyclic patterns in the time series. This requires us to calculate the weekly/daily cyclic average. Formally, let us denote $x(t)$ as the *raw traffic series*. We estimate the moving average of this series based on the same time of the same day of the week, *i.e.*,

$$\bar{x}(t) = \sum_{i=1}^{W} x(t - 24 \times 7 \times i)/W \tag{IV.8}$$

where W is the size of moving window. To eliminate the effect of bursty traffic, we also filter out the spike traffic during the above averaging procedure. Fig. IV.16(a)

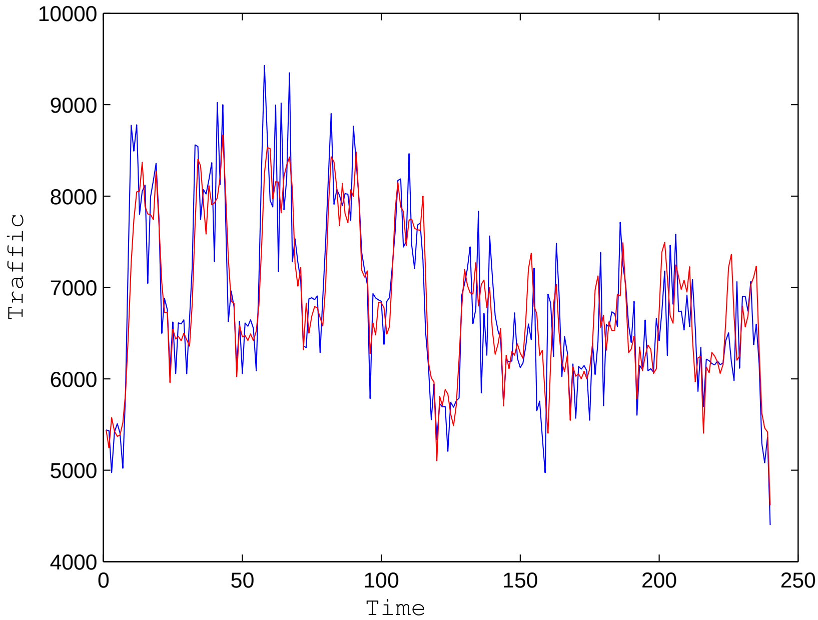


Figure IV.12: Predicted Traffic for the Original Series

plots the raw traffic as well as its moving average with $W = 5$. By removing the cyclic effect from the raw data, we derive the *adjusted traffic series* $z(t)$ as follows.

$$z(t) = x(t) - \bar{x}(t) \tag{IV.9}$$

The adjusted series of the one shown in Fig. IV.16(a) is given in Fig. IV.16(b). This adjusted traffic exhibits short-term (a few hours) traffic correlations. We model the adjusted traffic series with an autoregressive process as follows[1].

$$z(t) = \beta_1 z(t-1) + \beta_2 z(t-2) + ... + \beta_K z(t-K) + \epsilon \tag{IV.10}$$

where K is the process order. To apply this model for prediction, we estimate the parameters of this process. Given N observations $z_1, z_2, ..., z_N$, the parameters β_1,

[1]Ideally, $z(t)$ should have zero mean. In some cases, $z(t)$ has a small mean value which needs to be removed before fitting an autoregressive process.

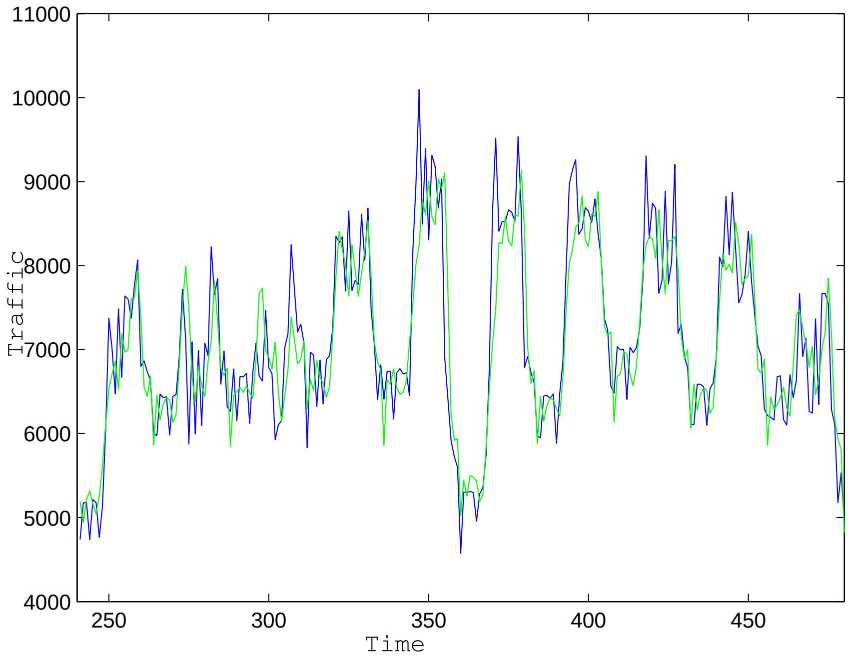


Figure IV.13: Raw Prediction Based Traffic without Seasonality

..., β_K are estimated via least squares by minimizing:

$$\sum_{t=K+1}^{N} \left[z(t) - \beta_1 z(t-1) ... - \beta_K z(t-K) \right]^2 \tag{IV.11}$$

Based on these parameters, we further derive the adjusted traffic prediction $\hat{z}(t)$ as follows:

$$\hat{z}(t) = \beta_1 z(t-1) + \beta_2 z(t-2) + ... + \beta_K z(t-K) \tag{IV.12}$$

Fig. IV.17 illustrates the estimation results for the adjusted traffic series in Fig. IV.16(b), where $K = 2$, $\beta_1 = 0.531$, $\beta_2 = 0.469$. The figure plots the predicted series for the adjusted traffic as well as its raw data. In this figure, the number

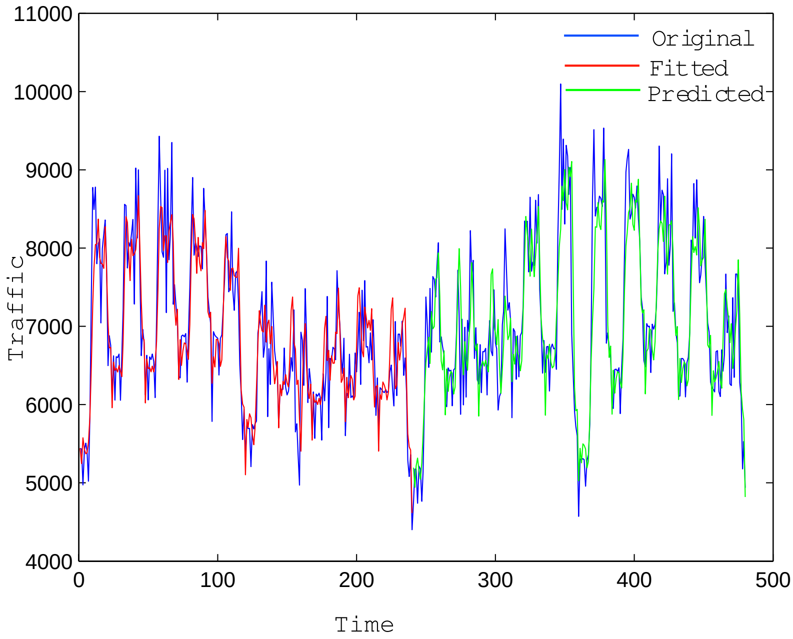


Figure IV.14: Predicted Traffic for the Original Series

of observations used for parameter estimation is $N = 60$. For the purpose of comparison, the fitted traffic series is also plotted for the interval $[720, 779]$.

We now consider the errors involved in this prediction process. In particular, we define the adjusted traffic prediction error as follows.

$$\epsilon_z(t) = z(t) - \hat{z}(t) \tag{IV.13}$$

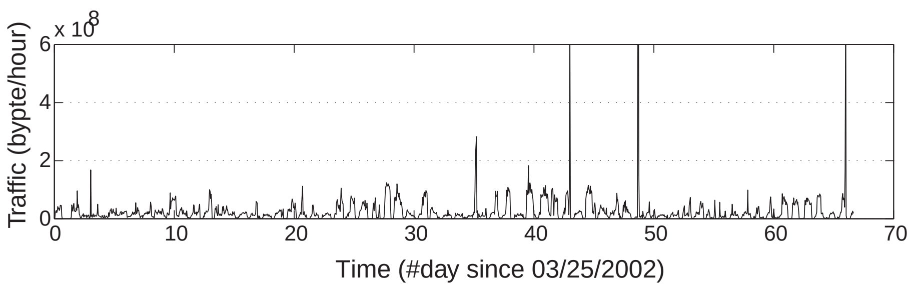
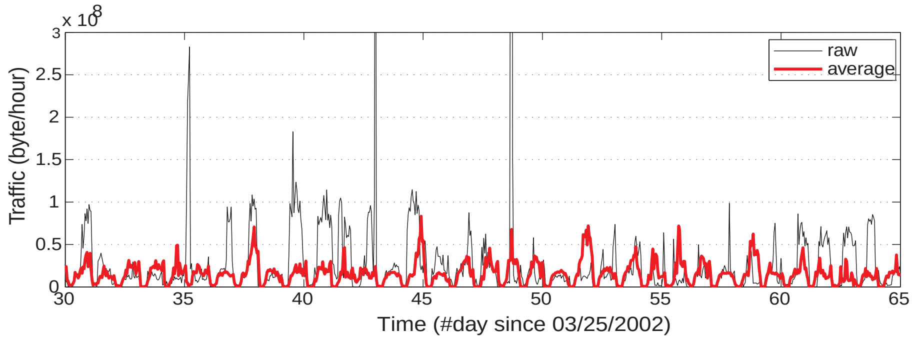
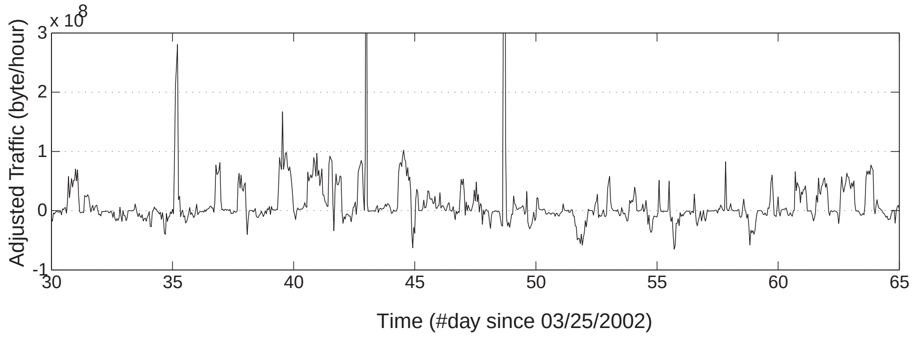


Figure IV.15: Incoming Traffic Time Series of A Residential Building on an Academic Campus



(a) Raw Traffic vs. Moving Average Series



(b) Adjusted Traffic Series

Figure IV.16: Traffic Series in 5 weeks

Based on this definition, Fig. IV.18(a) plots the cumulative distribution function of the prediction error of the adjusted traffic series shown in Fig. IV.17. It is obvious that the error distribution fits the normal distribution with a mean close to zero.

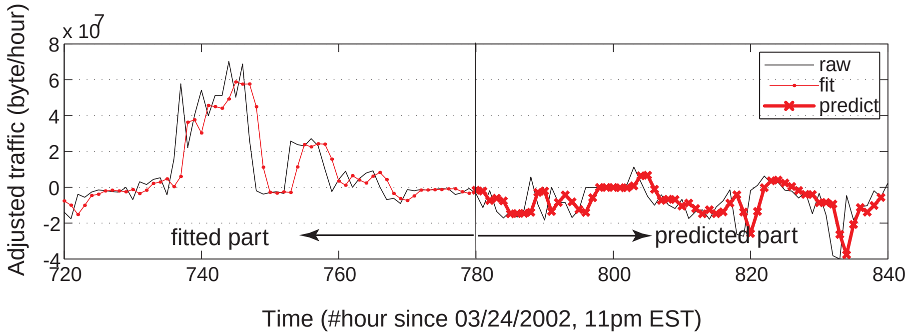
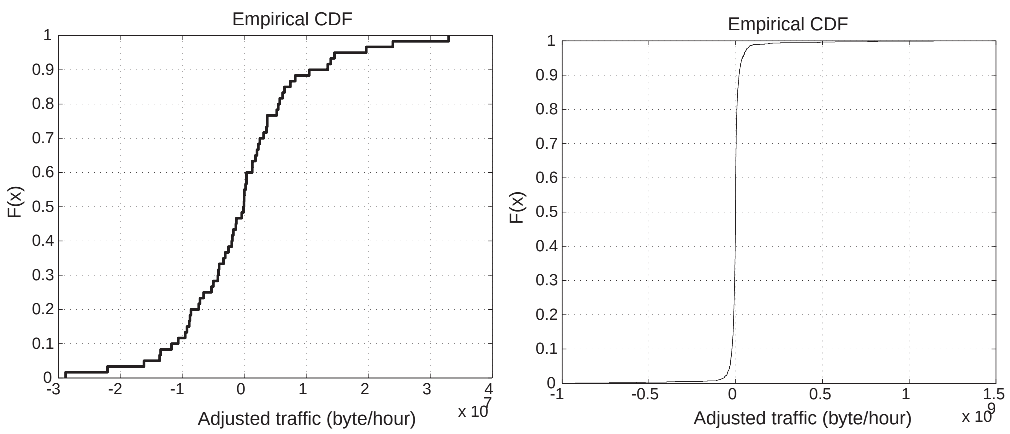


Figure IV.17: Adjusted Traffic and Its Prediction

Finally, we define traffic prediction $\hat{x}$ as follows:

$$\hat{x}(t) = [\bar{x}(t) + \hat{z}(t)]^{+} \qquad \text{(IV.14)}$$

where $[x]^{+} = \max\{0, x\}$. Fig. IV.19 plots the predicted traffic series $\hat{x}(t)$ in comparison with the raw traffic. We can see that the predicted traffic closely matches the real(raw) traffic. The cumulative distribution function of the prediction error $\epsilon_x(t)$, which is defined as $\epsilon_x(t) = x(t) - \hat{x}(t)$, is plotted in Fig. IV.18(b). It clearly shows that this distribution also fits the normal distribution with a near-zero mean.



(a) Prediction error for adjusted traffic (b) Prediction error for entire series

Figure IV.18: Cumulative Density Function of Prediction Error

We can consider the estimated traffic demand at time t as a random variable $X(t)$ which follows the normal distribution with mean $\hat{x}(t)$ and the same variance as ϵ_x. Fig. IV.20 shows the distribution of the predicted traffic demand of the 976th hour.

To summarize, the presented traffic prediction method provides two traffic models: mean value and statistical distribution. These two traffic models will serve as

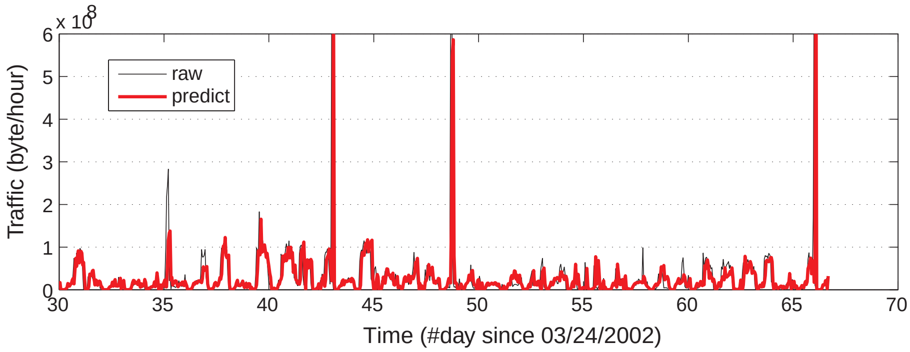


Figure IV.19: Raw Traffic vs. Predicted Traffic

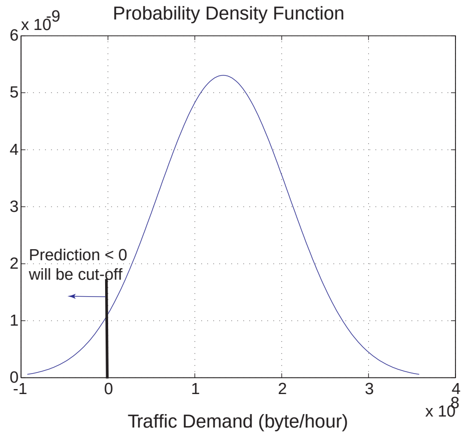


Figure IV.20: Traffic Estimation Distribution

the inputs for the fixed-demand mesh network routing algorithm (**FMR**) and the uncertain-demand mesh network routing algorithm (**UMR**), which are presented in the next chapter.

CHAPTER V

ROUTING ALGORITHMS FOR SINGLE-CHANNEL WIRELESS MESH NETWORKS

Solution Overview

This chapter presents an integrated framework for single-channel wireless mesh network routing, which integrates the demand prediction into traffic routing so that minimum congestion will be incurred. This routing objective can be transformed into the throughput optimization problem, where the throughput of aggregated flows is maximized subject to fairness constraints that are weighted by the traffic demands. In particular, two forms of traffic demands are considered as the inputs for routing optimization, namely the *mean value* of the demand prediction and its *statistical distribution*. We present two routing algorithms for each form of the traffic demand estimation respectively. For the first case, based on the classical maximum concurrent flow problem, we formulate optimal mesh network routing as a linear programming problem to maximize, among all flows, the minimum scaling factor of throughput to fixed-value demand (λ) and present a fast $(1-\epsilon)$-approximation algorithm (*i.e.* fixed-demand mesh network routing (**FMR**) algorithm) which could accept the mean value of the demand prediction as the input. For the second case, in order to incorporate the statistical distribution of the demand estimation into the problem formulation, we characterize the traffic demand using a random variable. Now the scaling factor λ under a given routing solution is also a random variable. The throughput optimization problem is then extended to a stochastic optimization problem where the expected value of the scaling factor λ is maximized. Finally, based on the design of **FMR** algorithm, a $(1-\epsilon)$-approximation algorithm (uncertain-demand mesh network routing (**UMR**)) is presented for optimal mesh network routing under uncertain demand.

The notations used in this chapter are summarized in Table V.1.

Notation	**Definition**
$G = (V, E)$	Network
$G' = (V', E')$	Network with virtual gateway/links
$u \in V$	Node
$e = (u, v) \in E$	Edge connecting nodes u and v
$f \in F$	Aggregated flow
$\boldsymbol{x} = (x_f, f \in F)$	Aggregated flow rate vector
$\boldsymbol{y} = (y_e, e \in E)$	Wireless link rate vector
$\boldsymbol{d} = (d_f, f \in F)$	Flow traffic demands
$p(\boldsymbol{d})$	Probability of $\boldsymbol{d}$
$\mathcal{P}_f$	Set of paths that can route f
$x_f(P)$	Rate of flow f over path $P \in \mathcal{P}_f$
S_e	Adjusted interference set of $e \in E$
$A_{eP} = \lvert S_e \cap P \rvert$	Number of wireless links P passes in S_e
$x(t)$	Raw traffic series
$z(t)$	Adjusted traffic series
$\bar{x}(t)$	Average traffic series
$\hat{x}(t)$	Predicted traffic series
$\hat{z}(t)$	Predicted adjusted traffic series
$\epsilon_x(t), \epsilon_z(t)$	Prediction error
$\lambda = \min_{f \in F}\{\frac{x_f}{d_f}\}$	Scaling factor
$\theta = \max_{e \in E}\{\frac{\sum_{e' \in S_e} y'_e}{c}\}$	Congestion, maximum adjusted independent set utilization
μ_e	Price of S_e

Table V.1: Notations

Fixed Demand Mesh Network Routing

This section investigates the optimal routing strategy for wireless mesh backbone networks under fixed traffic demand. A common routing performance metric with respect to a fixed traffic demand is *resource utilization.* For example, link utilization is commonly used for traffic engineering in the Internet [101], whose objective is to minimize the utilization at the most congested link. However, in a multihop wireless network, such as a mesh backbone network, wireless link utilization may be inappropriate as a metric of routing performance due to the location-dependent interference.

On the other hand, the existing works on optimal mesh network routing [18] usually aim at maximizing the flow throughput, while satisfying the fairness constraints. In this formulation, traffic demand is reflected as the flow weight in the fairness constraints. In light of these results, we first outline the relation between the throughput optimization problem and the congestion minimization problem, and define the utilization (so-called *congestion*) of the adjusted interference set as the routing performance metric. We show that the solution derived from the throughput optimization could naturally lead to the routing scheme which balances the resource utilization and minimizes the network congestion under fixed traffic demand. We then present a fully polynomial time approximation algorithm, which finds an ϵ-approximate solution. The problem formulation and algorithm presented in this section will accept the mean-value traffic prediction as the input for routing. It also serves as the basis of uncertain demand routing discussed in the next sections.

Problem Formulation

We first study the formulation of the throughput optimization routing problem in a wireless mesh backbone network under the fixed traffic demand. We regard the virtual node w^* that connects to gateways as the source of all incoming traffic and the destination of all outgoing traffic of a mesh network. Similarly, the local access points, which aggregate the client traffic, serve as the sources of all outgoing traffic and the destinations of incoming traffic. It is worth noting that although we consider only the aggregated traffic between gateway access points and local access points, our problem formulations and algorithms could be easily extended to handle inter-mesh-router traffic. Recall that $f \in F$ is the aggregated traffic flow between the local access points and the virtual gateway. We use d_f to denote the demand of flow f and $\boldsymbol{d} = (d_f, f \in F)$ to denote the demand vector consisting of all flow demands. Consider the fairness constraint that, for each flow f, its throughput being routed is in proportion to its demand d_f. Our goal is to maximize λ (so called *scaling factor*) where at least $\lambda \cdot d_f$ amount of throughput can be routed for flow f. We assume an infinitesimally divisible flow model where the aggregated traffic flow could be routed over multiple paths and use $\mathcal{P}_f$ to denote the set of unicast paths that could route flow f.

Let $x_f(P)$ be the rate of flow f over path $P \in \mathcal{P}_f$. Obviously the aggregated flow rate y_e along edge $e \in E$ is given by $y_e = \sum_{f:P\in\mathcal{P}_f \& e\in P} x_f(P)$, which is the sum of the flow rates that are routed through paths P passing edge e. Based on the sufficient condition of schedulability in Claim 1 (Eq.(III.1)), we have that

$$\sum_{e'\in S_e} \sum_{f:P\in\mathcal{P}_f \& e'\in P} x_f(P) \leq c \tag{V.1}$$

To simplify the above equation, we define $A_{eP} = |S_e \cap P|$ as the number of wireless links path P passes in the adjusted interference set S_e. The throughput optimization routing with fairness constraint is then formulated as the following linear programming (LP) problem:

$$\begin{aligned}
\mathbf{P_T}: \quad & \text{maximize} \quad \lambda && \text{(V.2)}\\
& \text{subject to} \quad \sum_{P \in \mathcal{P}_f} x_f(P) \geq \lambda \cdot d_f, \forall f \in F && \text{(V.3)}\\
& \sum_{f \in F} \sum_{P \in \mathcal{P}_f} x_f(P) A_{eP} \leq c, \forall e \in E && \text{(V.4)}\\
& \lambda \geq 0, x_f(P) \geq 0, \forall f \in F, \forall P \in \mathcal{P}_f && \text{(V.5)}
\end{aligned}$$

In this problem, the optimization objective is to maximize λ, such that at least $\lambda \cdot d_f$ units of data can be routed for each aggregated flow f with demand d_f. Inequality (V.3) enforces fairness by requiring that the comparative ratio of traffic routed for different flows satisfies the comparative ratio of their demands. Inequality (V.4) enforces capacity constraint by requiring the traffic aggregation of all flows passing wireless link $e \in E$ satisfy the sufficient condition of schedulability. This problem formulation follows the classical maximum concurrent flow problem.

Now we proceed to study the congestion minimization routing. Let $x'_f(P)$ be the rate of flow f on path P under traffic demand d_f. It is obvious that $\sum_{P \in P_f} x'_f(P) = d_f$. The traffic being routed within the adjusted interference set S_e is given by $\sum_{f \in F} \sum_{P \in \mathcal{P}_f} x'_f(P) A_{eP}$. We define the *congestion* of an adjusted interference set S_e using its utilization (*i.e.*, the ratio between its load and the channel capacity) and denote it as θ_e:

$$\theta_e = \frac{\sum_{f \in F} \sum_{P \in \mathcal{P}_f} x'_f(P) A_{eP}}{c} \tag{V.6}$$

Further, we define $\theta = \max_{e \in E} \theta_e$ as the maximum congestion among all the adjusted interference sets. The congestion minimization routing problem is then formulated as follows:

$$
\begin{aligned}
\mathbf{P_C}: \quad & \text{minimize} \quad \theta && \text{(V.7)}\\
& \text{subject to} \quad \sum_{P \in \mathcal{P}_f} x'_f(P) \geq d_f, \forall f \in F && \text{(V.8)}\\
& \sum_{f \in F} \sum_{P \in \mathcal{P}_f} x'_f(P) A_{eP} \leq c \cdot \theta, \forall e \in E && \text{(V.9)}\\
& \theta \geq 0, x'_f(P) \geq 0, \forall f \in F, \forall P \in \mathcal{P}_f && \text{(V.10)}
\end{aligned}
$$

To reveal the relation between $\mathbf{P_T}$ and $\mathbf{P_C}$, we let $\theta = \frac{1}{\lambda}$ and $x'_f(p) = \frac{x_f(p)}{\lambda}$. Problem $\mathbf{P_C}$ is then transformed to:

$$
\begin{aligned}
\mathbf{P'_C}: \quad & \text{minimize} \quad \frac{1}{\lambda} && \text{(V.11)}\\
& \text{subject to} \quad \frac{1}{\lambda} \sum_{P \in \mathcal{P}_f} x_f(P) \geq d_f, \forall f \in F && \text{(V.12)}\\
& \frac{1}{\lambda} \sum_{f \in F} \sum_{P \in \mathcal{P}_f} x'_f(P) A_{eP} \leq c \cdot \theta, \forall e \in E && \text{(V.13)}\\
& \lambda \geq 0, x'_f(P) \geq 0, \forall f \in F, \forall P \in \mathcal{P}_f && \text{(V.14)}
\end{aligned}
$$

which is obviously equivalent to the throughput optimization problem $\mathbf{P_T}$.

Algorithm

Both problems $\mathbf{P_T}$ and $\mathbf{P_C}$ could be solved by an LP-solver such as [102]. To reduce the complexity for practical use, we present a fully polynomial time approximation algorithm for problem $\mathbf{P}_T$, which finds an ϵ-approximate solution. The key to a fast approximation algorithm lies on the dual of this problem, which is formulated as follows. We assign a price μ_e to each set S_e for $e \in E$. The objective is to minimize the aggregated price for all adjusted interference sets. As the constraint, Inequality (V.16) requires that the price $\sum_{e\in E} A_{eP}\mu_e$ of any path $P \in \mathcal{P}_f$ for flow f must be at least μ_f, the price of flow f. Further, Inequality (V.17) requires that the weighted flow price μ_f over its demand d_f must be at least 1.

$$\mathbf{D_T}: \quad \text{minimize} \quad \sum_{e\in E} c\cdot\mu_e \tag{V.15}$$

$$\text{subject to} \quad \sum_{e\in E} A_{eP}\mu_e \geq \mu_f, \forall f \in F, \forall P \in \mathcal{P}_f \tag{V.16}$$

$$\sum_{f\in F} \mu_f d_f \geq 1 \tag{V.17}$$

FMR: Mesh Network Routing Under Fixed Demand

$\forall e \in E$, $\mu_e \leftarrow \beta/c$
$x_f(P) \leftarrow 0$, $\forall P \in \mathcal{P}_f$, $\forall f \in F$
while $\sum_{e\in E} c\cdot\mu_e < 1$
  for $\forall f \in F$ **do**
    $d'_f \leftarrow d_f$
    while $\sum_{e\in E} c\cdot\mu_e < 1$ **and** $d'_f > 0$ **do**
      $P \leftarrow$ lowest priced path in $\mathcal{P}_f$ using μ_e
      $\delta \leftarrow \min\{d'_f, \min_{e\in P} A_{eP}\}$
      $d'_f \leftarrow d'_f - \delta$
      $x_f(P) \leftarrow x_f(P) + \delta$
      $\forall e$ s.t. $A_{eP} \neq 0$, $\mu_e \leftarrow \mu_e(1 + \epsilon\delta A_{eP})$
    end while
  end for
end for

Table V.2: Routing Algorithm Under Fixed Demand

Based on the above dual problem $\mathbf{D_T}$, our fast approximation algorithm is presented in Table V.2. The algorithm design follows the idea of [59]. To start, we initialize the price on each adjusted interference set S_e as β/c (Line 1). We also zero the traffic on all paths $P \in \mathcal{P}_f$ (Line 2). Then for each flow f, we route d_f units of data. We do so by finding the lowest priced path in the path set $\mathcal{P}_f$ (Line 7), then filling traffic to this path by its bottleneck capacity (Lines 8 to 10). Then we update the prices for adjusted interference sets appeared in this path based on the function defined in Line 11. We keep filling traffic to flow f in the above fashion until all d_f units are routed. This procedure is repeated until the aggregate price of interference sets S_e for all $e \in E$ weighted by c exceeds 1 (Line 3).

We make following notes to our algorithm. First, it completes in finite time, which is guaranteed by the asymptotic link price update function defined in Line 11. ϵ here is the step size, which controls the growing speed of the link price. Second, since capacity c is the same for all the adjusted interference sets, its value does not affect the routing solution. Third, as one might see, the algorithm in fact routes more traffic than its actual demand, Therefore, a scaling procedure is needed to scale down the routed traffic so it fits its actual demand. In particular, $x_f(P)$ will be scaled as follows

$$x'_f(P) = x_f(P) \cdot \frac{d_f}{\sum_{P \in \mathcal{P}_f} x_f(P)} \tag{V.18}$$

We formally analyze the properties of our algorithm in the following theorem. The proofs of the theorems in this chapter are available in the Appendix I.

Theorem 1: If $\beta = (|E|/(1-\epsilon))^{-1/\epsilon}$, then the final flow generated by **FMR** is at least $(1-3\epsilon)$ times the optimal value of $\mathbf{P}$. The running time is $O(\frac{1}{\epsilon^2}[\log|E|(2|F|\log|F|+|E|)+\log U)]) \cdot T_{mp}$, where U is the length of the longest path in G, and T_{mp} is the running time to find the shortest path.

The actual time complexity in terms of $|V|$ can be analyzed as follows. The time complexity for the shortest path routing algorithm T_{mp} is usually $O(|V|^2)$. U, on the other hand, represents the longest path in G and should have a time complexity no larger than $O(|V|)$. The same answer can be applied to $|F|$. Although $|F|$ can be much less than $|V|$, the upper bound can be no larger than $|V|$. We have the same issue in estimating $|E|$. Although $|E|$ can vary from a small number to a very large number, it will never be over its upper bound of $|V|^2$. Following previous reduction, we have a simplified upper bond of $|V|^4 \log |V|$. However, we should realize that it is a much loose upper bound, and the actual time complexity should be lower than this value.

Uncertain Demand Mesh Network Routing

Now we proceed to investigate the throughput optimization routing problem for wireless mesh backbone network when the aggregated traffic demand is uncertain. We model such uncertain traffic demand of an aggregated flow $f \in F$ using a random variable D_f. We assume that D_f follows the following discrete probability distribution

$$Pr(D_f = d_f^i) = q_f^i \tag{V.19}$$

where $\mathcal{D}_f = \{d_f^1, d_f^2, ..., d_f^m\}$ is the set of of values for D_f with non-zero probabilities. Let $\boldsymbol{d} = (d_f, d_f \in \mathcal{D}_f, f \in F)$ be a sample traffic demand vector, $\boldsymbol{D}$ be the corresponding random variable, and $\mathcal{D}$ be the sample space. We further assume that the demand from different access points are independent from each other. Thus the distribution of $\boldsymbol{D}$ is given by the joint distribution of these random variables as follows.

$$Pr(\boldsymbol{D} = \boldsymbol{d}) = Pr(D_f = d_f^i, f \in F) = \Pi_{f \in F} q_f^i \tag{V.20}$$

Let us consider a traffic routing solution $(x_f(P), P \in \mathcal{P}_f, f \in F)$ that satisfies the capacity constraint (Inequality (V.4)). It is obvious that λ is a function of $\boldsymbol{d}$:

$$\lambda(\boldsymbol{d}) = \min_{f \in F} \{\frac{x_f}{d_f}\} \tag{V.21}$$

where $x_f = \sum_{P \in \mathcal{P}_f} x_f(P)$. Further let us consider the optimal routing solution under demand vector $\boldsymbol{d}$. Such a solution could be easily derived based on Algorithm we shown in Table V.2. We denote the optimal value of λ as $\lambda^*(\boldsymbol{d})$. We further define the *performance ratio* ω of routing solution $(x_f(P), P \in \mathcal{P}_f, f \in F)$ as follows:

$$\omega(\boldsymbol{d}) = \frac{\lambda(\boldsymbol{d})}{\lambda^*(\boldsymbol{d})}$$

Obviously, the performance ratio is also a random variable under uncertain demand. We denote it as Ω. Ω is a function of random variable $\boldsymbol{D}$. Now we extend the wireless mesh network routing problem to handle such uncertain demand. Our goal is to maximize the expected value of Ω, which is given as follows.

$$E(\Omega) = Pr(\boldsymbol{D} = \boldsymbol{d}) \times \frac{\lambda(\boldsymbol{d})}{\lambda^*(\boldsymbol{d})} \tag{V.22}$$

We abbreviate $Pr(\boldsymbol{D} = \boldsymbol{d})$ as $p(\boldsymbol{d})$. It is obvious that $\sum_{d \in \mathcal{D}} p(\boldsymbol{d}) = 1$. Formally, we formulate the throughput optimization routing problem for wireless mesh backbone network under uncertain traffic demand as follows.

$$
\begin{aligned}
\mathbf{P_U}: \quad & \text{maximize} && \sum_{\boldsymbol{d}\in\mathcal{D}} p(\boldsymbol{d}) \frac{\lambda(\boldsymbol{d})}{\lambda^*(\boldsymbol{d})} && \text{(V.23)}\\
& \text{subject to} && \forall \boldsymbol{d} \in \mathcal{D}, \text{where } \boldsymbol{d} = (d_f, f \in F) && \\
& && \sum_{P\in\mathcal{P}_f} x_f(P) \geq \lambda(\boldsymbol{d}) \cdot d_f, \forall f \in F && \text{(V.24)}\\
& && \sum_{f\in F} \sum_{P\in\mathcal{P}_f} x_f(P) A_{eP} \leq c, \forall e \in E && \text{(V.25)}\\
& && \lambda \geq 0, x_f(P) \geq 0, \forall f \in F, \forall P \in \mathcal{P}_f && \text{(V.26)}
\end{aligned}
$$

Similar to problem $\mathbf{P_T}$, the constraints of $\mathbf{P_U}$ come from the fairness requirement and the wireless mesh network capacity. In particular, Inequality (V.24) enforces fairness for all demand $\boldsymbol{d} \in \mathcal{D}$, and Inequality (V.25) enforces capacity constraint as Inequality (V.4) in problem $\mathbf{P_T}$.

Now we consider the dual problem $\mathbf{D_U}$ of $\mathbf{P_U}$. Similar to $\mathbf{D_T}$, the objective of $\mathbf{D_U}$ is to minimize the aggregated price for all adjusted interference sets. However, in Inequality (V.29), for each sample demand vector $\boldsymbol{d}$, the aggregated price of all flows weighted by their demand needs to be larger than its probability.

$$
\begin{aligned}
\mathbf{D_U}: \quad & \text{minimize} && \sum_{e\in E} c \cdot \mu_e && \text{(V.27)}\\
& \text{subject to} && \sum_{e\in E} A_{eP}\mu_e \geq \mu_f, \forall f \in F, \forall P \in \mathcal{P}_f && \text{(V.28)}\\
& && \sum_{f\in F} \mu_f d_f \geq \frac{p(\boldsymbol{d})}{\lambda^*(\boldsymbol{d})}, \forall \boldsymbol{d} \in \mathcal{D} && \text{(V.29)}\\
& && \text{where } \boldsymbol{d} = (d_f, f \in F) &&
\end{aligned}
$$

UMR: Mesh Network Routing Under Uncertain Demand

$\forall e \in E$, $\mu_e \leftarrow \beta$
$x_f(P) \leftarrow 0$, $\forall P \in \mathcal{P}_f$, $\forall f \in F$
loop
  for $\forall f \in F$ **do**
    $\bar{P} \leftarrow$ lowest priced path in $\mathcal{P}_f$ using μ_e
    $\mu_f \leftarrow \sum_{e \in E} A_{e\bar{P}} \mu_e$
  end for
  for $\forall \boldsymbol{d} \in \mathcal{D}$ **do**
    $\mu_{\boldsymbol{d}} \leftarrow \sum_{f \in F} \mu_f d_f \frac{\lambda^*(\boldsymbol{d})}{p(\boldsymbol{d})}$
  end for
  $\mu^{\min} \leftarrow \min_{\boldsymbol{d} \in \mathcal{D}} \mu_{\boldsymbol{d}}$
  $\boldsymbol{d}^{\min} \leftarrow \arg\min_{\boldsymbol{d} \in \mathcal{D}} \mu^{\min}$
  if $\mu^{\min} \geq 1$
    return
  for $\forall f \in F$ **do**
    $d'_f \leftarrow d^{\min}_f$
    while $d'_f > 0$ **do**
      $P \leftarrow$ lowest priced path in $\mathcal{P}_f$ using μ_e
      $\delta \leftarrow \min\{d'_f, \min_{e \in P} \frac{1}{A_{eP}}\}$
      $d'_f \leftarrow d'_f - \delta$
      $x_f(P) \leftarrow x_f(P) + \delta$
      $\forall e$ s.t. $A_{eP} \neq 0$, $\mu_e \leftarrow \mu_e(1 + \epsilon\delta A_{eP} \times \frac{\lambda^*(\boldsymbol{d}^{\min})}{p(\boldsymbol{d}^{\min})})$
    end while
  end for
end loop

Table V.3: Routing Algorithm Under Uncertain Demand

Now we present an approximation algorithm for $\mathbf{P_U}$ in Table V.3. Note that since the channel capacity c will not affect the final result of the algorithm, we simply omit it here. This algorithm (**UMR**) has the same initialization as the algorithm for problem $\mathbf{P_T}$ (**FMR**). Then we march into the iteration, in which we find $\boldsymbol{d}^{\min}$, the demand whose price $\mu^{\min}$ is the minimum among others (Lines 4 to 12). If $\mu^{\min} \geq 1$, then the algorithm stops (Lines 13 and 14), since Inequality (V.28) and (V.29) would be satisfied for all demand. Otherwise, we will increase the price of $\boldsymbol{d}^{\min}$ by routing more traffic through its node pairs. This procedure (Lines 16 to 22) is the same as what has been described in Lines 4 to 11 of **FMR** algorithm. Following the same proving sequence for **FMR**, we am able to prove the similar properties with **UMR**.

Theorem 2: If $\beta = (|E|/(1-\epsilon))^{-1/\epsilon}$, then the final flow generated by **UMR** is at least $(1-3\epsilon)$ times the optimal value of $\mathbf{P_U}$. The running time is $O(\frac{1}{\epsilon^2}[\log|E|(2|\mathcal{D}||T_{fmr}||F|\log|E|) + \log U)]) \cdot T_{mp}$, where U is the length of the longest path in G, T_{mp} is the running time to find the shortest path, and T_{fmr} is the running time of the **FMR** algorithm.

Following the similar analysis in the previous algorithm, we can estimate the time complexity of this algorithm in terms of $|V|$. Since this algorithm is extended from **FMR**, it has a higher time complexity of about $|V|^7$

Simulation Study

Simulation Setup

AP	31AP3	34AP5	55AP4	57AP2	62AP3	62AP4	82AP4	94AP1	94AP3	94AP8
Node ID	22	18	57	5	55	20	53	3	56	27
AP	27AP3	3AP3	21AP2	23AP4	33AP2	62AP2	82AP3	84AP1	90AP2	97AP2
Node ID	9	23	25	33	19	35	58	42	6	48

Table V.4: Overview of Traffic Demand

We evaluate the performance of our algorithms via simulation study. In the simulated wireless mesh network, 60 mesh nodes are randomly deployed over a $1000 \times 2000m^2$ region. The simulated network topology is shown in Fig. V.1. In the basic setting, 10 nodes at the edge of this network are selected as the local access points (LAP) that forward traffic for clients. Two nodes (31 and 1) in the center of the deploy region are selected as the gateway access points. Aside from this basic setting, we have also evaluated the performance of our algorithms with different configurations of LAPs and gateways, which we will show at the later part of this section. Each mesh node has a transmission range of $250m$ and an interference range of $500m$. The channel capacity c is set as 54 Mbps.

To realistically simulate the traffic demand at each LAP, we employ the traces collected in the campus wireless LAN network. The network traces used in this work are collected in Spring 2002 at Dartmouth College and provided by CRAWDAD [100]. By analyzing the *snmp* log trace at each access point, we are able to derive its 1847-hour incoming and outgoing traffic volume since 12:00AM, March 25, 2002 EST. We argue that the LAPs of a wireless mesh network serve a similar

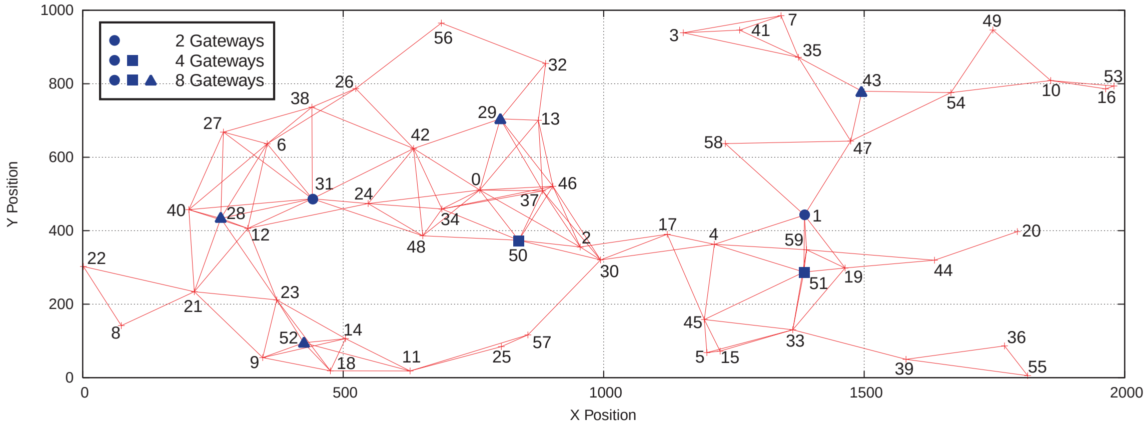


Figure V.1: Mesh Network Topology.

role as the access points of wireless LAN networks at aggregating and forwarding client traffic. Thus, we select the access points from the Dartmouth campus wireless LAN and assign their traffic traces to the LAPs in our simulation. The traffic assignment is given in Table V.4. In the basic setting, the 10 access points in the first row of the table are used.

We evaluate and compare different traffic prediction and routing strategies for this simulated network. In particular, we consider the following strategies.

- *Oracle Routing (OR)*. In this strategy, the traffic demand is known a priori. It runs the **FMR** algorithm (presented in Tab. V.2) based on this demand. This solution runs every hour based on the up-to-date traffic demand from the trace and returns the optimal set of routes. This ideal strategy is designed to return the benchmark result, which the rest of the practical strategies compare to.

- *Mean-Value Prediction Routing (MVPR)*. This strategy does not know the traffic demand a priori. Instead, it only predicts the traffic demand based on its historical data. In particular, it employs the mean value prediction model and runs the **FMR** algorithm based on this predicted demand. This solution also runs every hour to provide the set of routes for the next hour.

- *Statistical-Distribution Prediction Routing (SDPR)*. Similar to MVPR, this strategy also relies on traffic prediction. It predicts not only the mean-value of the traffic demand in the next hour, but also its distribution. It runs the **UMR** algorithm (presented in Tab. V.3) with the predicted traffic demand distribution as its input. Since **UMR** only accepts discrete probability distribution, we need to discretize the demand distribution as follows.

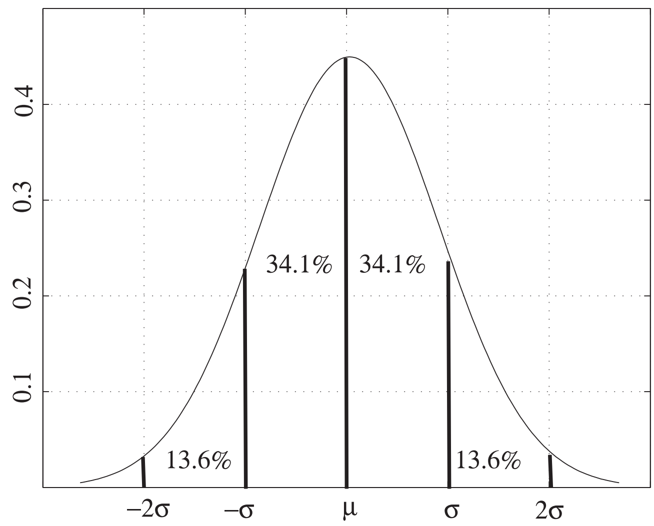


Figure V.2: Discretization of Traffic Distribution

As illustrated in Fig. V.2, we sample the following values, the mean value μ, and values $\mu - \sigma$, $\mu + \sigma$, $\mu - 2\sigma$, and $\mu - 2\sigma$. Since about 95% of all traffic demand values fall within the range $[\mu - 2\sigma, \mu + 2\sigma]$, we ignore the values which has a probability smaller than 5%.

- *Shortest-Path Routing (SPR)*. This strategy is agnostic of traffic demand, and returns a fixed routing solution purely based on the shortest distance (number of hops) from each mesh node to the gateway. The purpose to evaluate this strategy is to quantitatively contrast the advantage of our traffic-predictive routing strategies.

Simulation Results

We experiment with the above routing strategies along the time range $[108, 1847]$, a 1740-hour period excerpted from the trace[1]. We mainly study the congestion θ of each routing strategy under the given traffic demands.

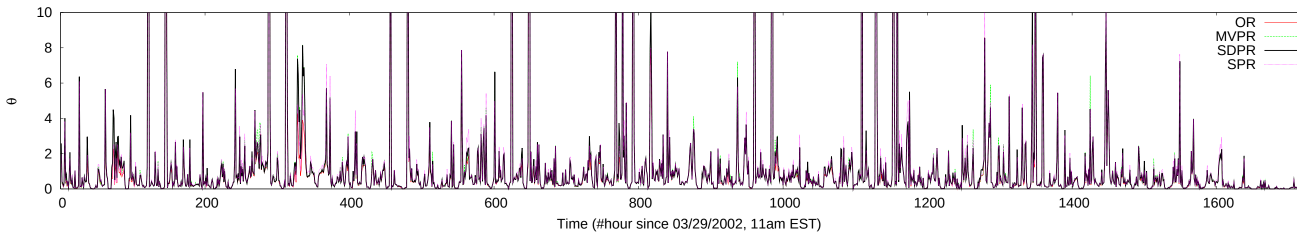
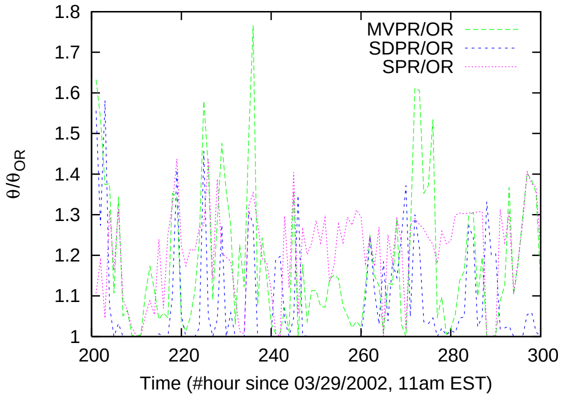


Figure V.3: Overview of All Strategies

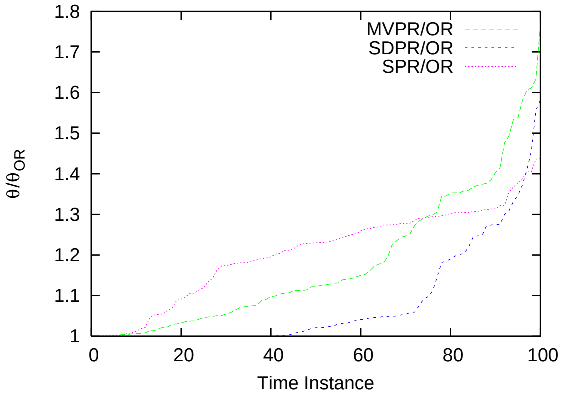
We start by presenting the congestion achieved by all strategies (*OR*, *MVPR*, *SDPR*, and *SPR*) during the entire 1740-hour simulation period. As seen in Fig. V.3, *OR* constantly achieves the minimum worst-case congestion among others, due to its unrealistic capability to know the actual traffic demand. We note that the burstiness of θ applies to all strategies including *OR*. Such observation comes from the burstiness of the traffic load in the *snmp* log trace, which is caused by the insufficient level of traffic multiplexing at wireless local access points.

To filter out the noise caused by traffic burstiness, in Fig. V.4(a), we normalize θ achieved by other strategies by the same value of *OR*. Since *OR* always achieves

[1]Note that the beginning part of the trace $[0, 107]$ is used as training data, thus is not included in the simulation result.



(a) Congestion Ratio



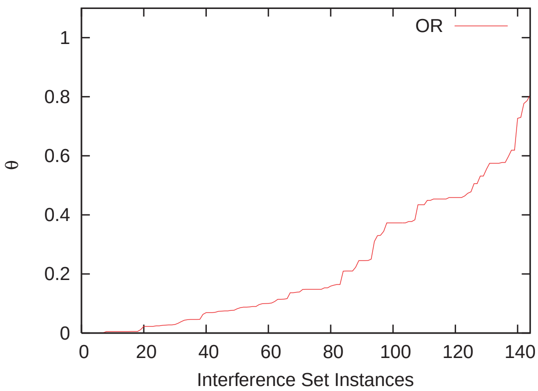
(b) Sorted Overview

Figure V.4: Comparison to OR

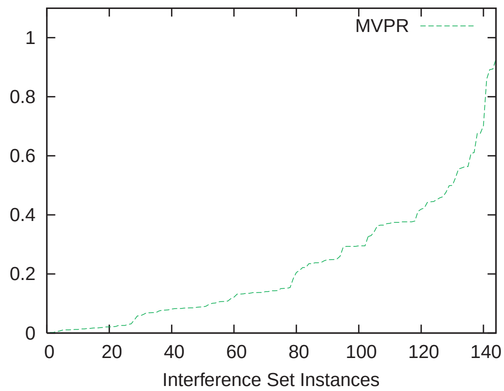
the minimum θ among others, this ratio will end up at least 1. Also we take a close-up look during the hour range $[201, 300]$. Here, all three strategies (*MVPR*, *SDPR*, and *SPR*) achieve less than 2 times of the optimal congestion. Although *MVPR* has the worst performance at a few occasions, *SPR* causes greater congestion than others in the most of the time, revealing the disadvantage of overlooking the varying traffic demand. *SDPR* constantly achieves lower congestion than *MVPR* due to more comprehensive representation of the traffic demand estimation.

The above observations get clearer when we sort out the normalized congestion ratio for the three strategies in Fig. V.4(b). Interestingly, although *SPR* is inferior to the traffic-prediction strategies generally, its worst-case congestion is lower than *MVPR* in 25% of the time, and *SDPR* in less than 5% of the time. This problem can be mostly attributed to the inaccuracy of traffic prediction. In other words, wrong estimation of traffic demand can cause routing solutions worse than being agnostic about it. However, more sophisticated prediction technique (*SDPR*) can greatly reduce its occurring probability than the simple one (*MVPR*).

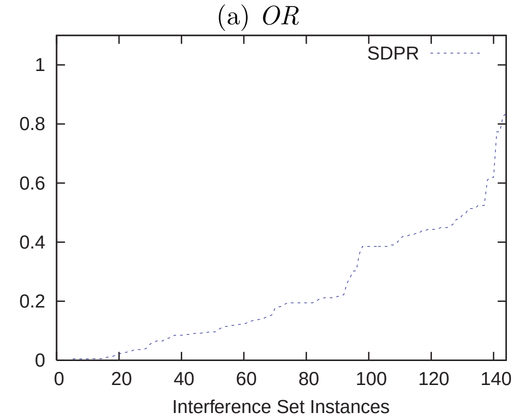
Next, we take a closer look at each strategy's ability to balance the traffic within the mesh network. In Fig. V.5, we unfold a single time instance at hour 1521 and exhibit the congestion at each adjusted interference set resulted from each strategy. In order to achieve the lowest worst-case congestion, a good strategy should maximally even out the traffic routed through all interference sets. Obviously, *OR* achieves such optimality, which resulting in the best θ value 0.8. *SPR* has the highest θ value as more than 1. The results for *MVPR* and *SDPR* are 0.9 and 0.8 respectively. In *MVPR*, 120 out of 140 interference sets have their congestions less than 0.4. Comparatively, in *SDPR*, about 100 interference sets are below this threshold, whereas the number is below 100 in *OR*, and above 120 in *SPR*. This observation keeps consistent when we repeat with different threshold in θ.



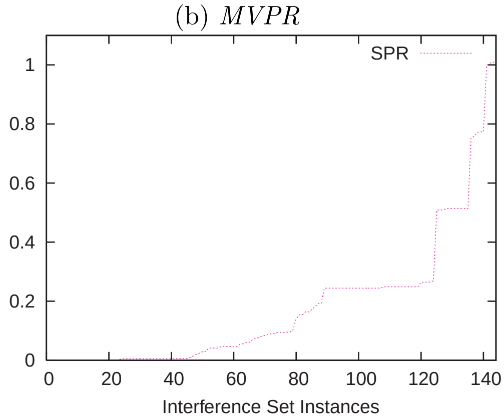
(a) *OR*



(b) *MVPR*

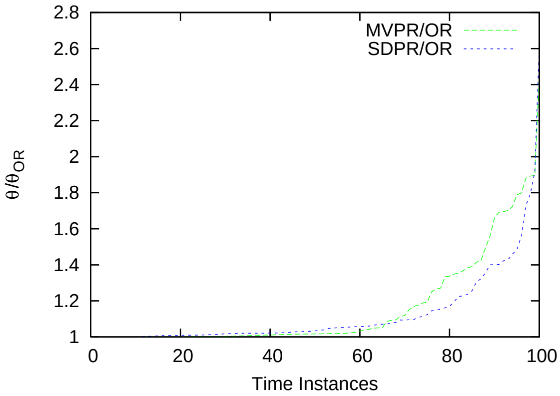


(c) *SDPR*

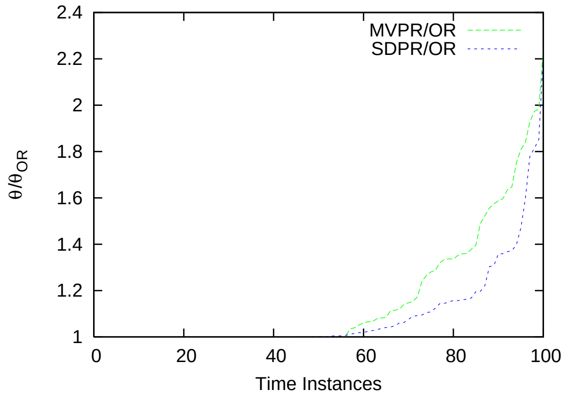


(d) *SPR*

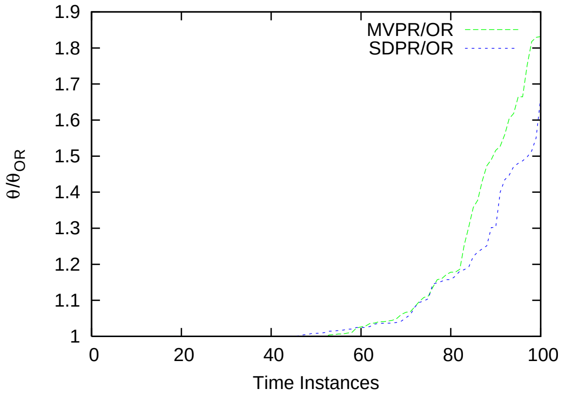
Figure V.5: Adjusted Interference Set Sorted By Congestion



(a) 2 Gateways



(b) 4 Gateways

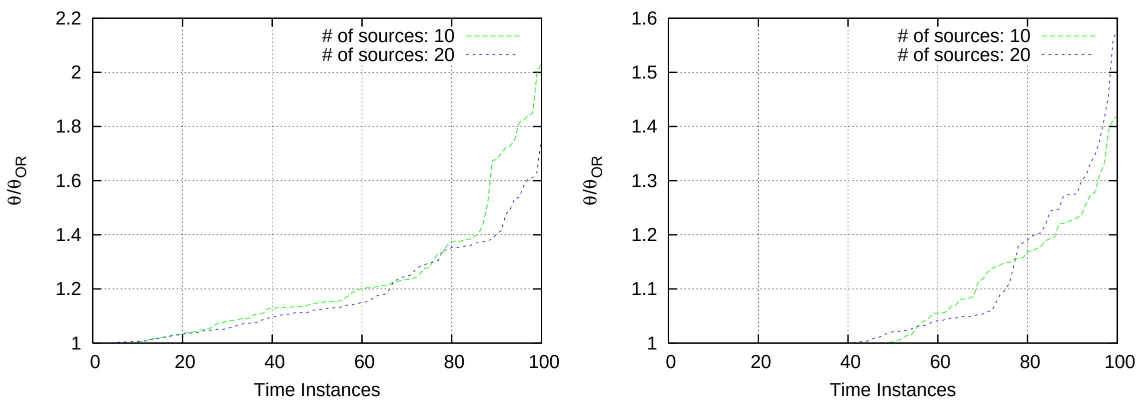


(c) 8 Gateways

Figure V.6: Impact by Number of Gateways

In what follows, we alter our simulation configurations to examine the abilities of different strategies at adapting various network settings. Here, we focus on the traffic prediction strategies, namely, *MVPR* and *SDPR*. Also we plot their performances by the congestion ratio θ/θ_{OR} normalized by the *OR* routing results.

Since deploying multiple gateways is a commonly-used solution to improve mesh network throughput and avoid hot spots, we first evaluate our solutions' capabilities at taking this advantage to reduce congestion. We emphasize the additionally-deployed gateways in Fig. V.1. In Fig. V.6, we observe that the highest congestion ratios by both strategies drop linearly as we double the number of gateways, i.e., 2.6 at 2 gateways, 2.2 at 4 gateways, and 1.8 at 8 gateways. Also *SDPR* consistently outperforms *MVPR* at approaching the optimal *OR* strategy. In case of 2 and 4 gateways, more than 80% of the time, the performance of *SDPR* is within 20% of optimal congestion, whereas the same value is 40% for *MVPR*. In addition, in case of 8 gateways, *SDPR* achieves within 60% of optimal congestion of all times, 20% less than *MVPR*.



(a) *MVPR* (b) *SDPR*

Figure V.7: Impact by Number of Sources

Finally, by doubling the number of LAPs to 20, we test our solutions' adaptability to traffic demand. The traffic assignments of these 20 LAPs are shown in Table V.4. Here, we observe that both solutions show good and stable approximation to the optimal *OR* strategy. In fact, more intensive traffic (with increasing multiplexing) makes our solution approximates closer to the optimal. Compared to the case of 10 LAPs, the worst-case performance of *MVPR* moves closer from within 100% optimality to 80%. For *SDPR*, it reduces from within 60% to 40%, consistently outperforming *MVPR* by 40%. In addition, *SDPR* achieves the same congestion with *OR* in 50% of the times, a 10% increase from the case of 10 LAPs.

CHAPTER VI

JOINT CHANNEL ASSIGNMENT AND ROUTING ALGORITHMS FOR MULTI-RADIO MULTI-CHANNEL WIRELESS MESH NETWORKS

Solution Overview

This chapter investigates the optimal routing strategy for wireless mesh *back-bone* networks with multi radios and multi channels. The solution expends the single-channel routing solution introduced in the previous chapter. As part of solution for multi-radio multi-channel WMN environment, we provide channel assignment solution in addition to optimal routing paths.

The performance of a multi-radio multi-channel wireless mesh network critically depends on the design of three interdependent components: scheduling, channel assignment, and routing. Their joint design has been studied in several existing works [38, 18]. In this chapter, we adopt the same approach as in [18] which formulates this problem as an integer linear programming problem. To solve this problem, [18] first solves its LP relaxation and derives the routing solution based on the necessary conditions of channel assignment and schedulability. Then the channel assignment and post processing algorithms are designed to adjust the flows to yield a feasible solution.

We assume that the system operates synchronously in a time-slotted mode. The result we obtain will provide an upper bound for systems using IEEE 802.11 MAC. We further assume that the traffic between a local access point and the Internet could be infinitesimally divided and routed over multiple paths to multiple gateways achieving the optimal load balancing and the least congestion.

The focus of this chapter is to investigate the optimal routing scheme under dynamic traffic based on the above necessary conditions of channel assignment and schedulability. Once the flow routes are derived, we simply apply the same method presented in [18] to adjust the flow routes and scale the flow rates to yield a feasible routing and channel assignment.

The objective is to determine the necessary and sufficient conditions for the link flow rates to be achievable in the network in terms of a valid schedule. We define a $0-1$ scheduling variable $Y_e(c)$

$$Y_e^t(c) = \begin{cases} 1 & \text{if link } e \text{ is active on channel } c \text{ in time slot } t \\ 0 & \text{otherwise} \end{cases}$$

The notations used in this chapter are summarized in Table VI.1.

<u>Fixed Demand Mesh Network Routing</u>

We first study the formulation of throughput optimization routing problem in a wireless mesh backbone network under the fixed traffic demand. We use d_f to denote the demand of flow f and $\boldsymbol{d} = (d_f, f \in F)$ to denote the demand vector consisting of all flow demands. Consider the fairness constraint that, for each flow f, its throughput being routed is in proportion to its demand d_f. Our goal is to maximize λ (so called *scaling factor*) where at least $\lambda \cdot d_f$ amount of throughput can be routed for flow f.

We assume an infinitesimally divisible flow model where the aggregated traffic flow could be routed over multiple paths and use $\mathcal{P}_f$ to denote the set of unicast paths that connect the source of f and w^*. Let $x_f(P)$ be the rate of flow f over path $P \in \mathcal{P}_f$. Obviously the link flow rate $y_e(c)$ is given by $y_e(c) =$

Notation	Definition
$G = (V, E)$	Network
$G' = (V', E')$	Network with virtual gateway/links
$u \in V$	Node
$e = (u, v) \in E$	Edge connecting nodes u and v
$f \in F$	Aggregated flow
$\boldsymbol{x} = (x_f, f \in F)$	Aggregated flow rate vector
$\boldsymbol{y} = (y_e, e \in E)$	Wireless link rate vector
$\boldsymbol{d} = (d_f, f \in F)$	Flow traffic demands
$p(\boldsymbol{d})$	Probability of $\boldsymbol{d}$
$\mathcal{P}_f$	Set of paths that can route f
$x_f(P)$	Rate of flow f over path $P \in \mathcal{P}_f$
S_e	Adjusted interference set of $e \in E$
$A_{eP} = \lvert S_e \cap P \rvert$	Number of wireless links P passes in S_e
$x(t)$	Raw traffic series
$z(t)$	Adjusted traffic series
$\bar{x}(t)$	Average traffic series
$\hat{x}(t)$	Predicted traffic series
$\hat{z}(t)$	Predicted adjusted traffic series
$\epsilon_x(t), \epsilon_z(t)$	Prediction error
$\lambda = \min_{f \in F}\{\frac{x_f}{d_f}\}$	Scaling factor
$\theta = \max_{e \in E}\{\frac{\sum_{e' \in S_e} y'_e}{c}\}$	Congestion, maximum adjusted independent set utilization
μ_e	Price of S_e

Table VI.1: Notations

$\sum_{f:P\in\mathcal{P}_f\&e(c)\in P} x_f(P)$, which is the sum of the flow rates that are routed through paths P passing edge $e(c) \in E_c$. Based on the necessary conditions of scheduling and channel assignment in Claim 1 (Eq.(III.3) and Eq.(III.4)), we have that

$$\sum_{e'(c)\in I_e(c)} \frac{1}{\phi_e(c)} \sum_{f:P\in\mathcal{P}_f\&e'(c)\in P} x_f(P) \leq \gamma(\Delta); \forall e(c) \in E_c \tag{VI.1}$$

$$\sum_{c\in C} \sum_{e'(c)\in E(v)} \frac{1}{\phi_e(c)} \sum_{f:P\in\mathcal{P}_f\&e'(c)\in P} x_f(P) \leq \kappa(v); \forall v \in V \tag{VI.2}$$

To simplify the above equations, we define $A_{e(c)P} = \sum_{e'(c)\in I_{e(c)}, e'(c)\in P} \frac{1}{\phi_{e'(c)}}$ and $B_{vP} = \sum_{c\in C} \sum_{e'(c)\in E(v), e'(c)\in P} \frac{1}{\phi_{e'(c)}}$. The throughput optimization routing with fairness constraint is then formulated as the following LP problem:

$$\begin{aligned}
\mathbf{P_T}: \quad & \text{maximize} \quad \lambda & \text{(VI.3)}\\
& \text{subject to} \quad \sum_{P\in\mathcal{P}_f} x_f(P) \geq \lambda \cdot d_f, \forall f \in F & \text{(VI.4)}\\
& \sum_{f\in F} \sum_{P\in\mathcal{P}_f} x_f(P) A_{e(c)P} \leq \gamma(\Delta), & \\
& \forall e(c) \in E_c & \text{(VI.5)}\\
& \sum_{f\in F} \sum_{P\in\mathcal{P}_f} x_f(P) B_{vP} \leq \kappa(v), \forall v \in V & \text{(VI.6)}\\
& \lambda \geq 0, x_f(P) \geq 0, \forall f \in F, \forall P \in \mathcal{P}_f & \text{(VI.7)}
\end{aligned}$$

In this problem, the optimization objective is to maximize λ, such that at least $\lambda \cdot d_f$ units of data can be routed for each aggregated flow f with demand d_f. Inequality (VI.4) enforces fairness by requiring that the comparative ratio of traffic routed for different flows satisfies the comparative ratio of their demands. Inequality (VI.5) and (VI.6) come from the necessary conditions of channel assignment and scheduling. This problem formulation follows the same form as the maximum concurrent flow problem.

Problem $\mathbf{P_T}$ could also be solved by a LP-solver [102]. To reduce the complexity for practical use, we present a fully polynomial time approximation algorithm for problem $\mathbf{P_T}$, which finds an ϵ-approximate solution. The key to a fast approximation algorithm lies on the dual of this problem, which is formulated as follows. We assign a price μ_e to each set $I_e(c)$ for $e(c) \in E_c$ and a price μ_v to each node $v \in V$. The objective is to minimize the aggregated price for all interference sets and all nodes. As the constraint, Inequality (VI.9) requires that the price $\sum_{e(c)\in E_c} A_{e(c)P}\mu_e + \sum_{v\in V} B_{vP}\mu_v$ of any path $P \in \mathcal{P}_f$ for flow f must be at least μ_f, the price of flow f. Further, Inequality (VI.10) requires that the weighted flow price μ_f over its demand d_f must be at least 1.

$$\mathbf{D_T}: \quad \text{minimize} \quad \sum_{e(c)\in E_c} \gamma(\Delta)\cdot\mu_e + \sum_{v\in V}\kappa(v)\mu_v \tag{VI.8}$$

$$\text{subject to} \quad \sum_{e(c)\in E_c} A_{e(c)P}\mu_e + \sum_{v\in V} B_{vP}\mu_v \geq \mu_f, \quad \forall f \in F, \forall P \in \mathcal{P}_f \tag{VI.9}$$

$$\sum_{f\in F}\mu_f d_f \geq 1 \tag{VI.10}$$

Based on the above dual problem $\mathbf{D_T}$, our fast approximation algorithm is presented in Table VI.2. The algorithm design follows the idea of [59]. In particular, Line 1 and Line 2 initialize the algorithm. Then for each flow f, we route d_f units of data. We do so by finding the lowest priced path in the path set $\mathcal{P}_f$ (Line 7),

FMR: Mesh Network Routing Under Fixed Demand	
1	$\forall e \in E$, $\gamma \leftarrow \gamma(\Delta)$, $\mu_e \leftarrow \beta/\gamma$, $\mu_v \leftarrow \beta/\kappa(v)$
2	$x_f(P) \leftarrow 0$, $\forall P \in \mathcal{P}_f$, $\forall f \in F$
3	**while** $\sum_{e(c)\in E(c)} \gamma \cdot \mu_e + \sum_{v\in V} \kappa(v)\mu_v < 1$
4	**for** $\forall f \in F$ **do**
5	$d'_f \leftarrow d_f$
6	**while** $\sum_{e(c)\in E(c)} \gamma \cdot \mu_e + \sum_{v\in V} \kappa(v)\mu_v < 1$ **and** $d'_f > 0$ **do**
7	$P \leftarrow$ lowest priced path in $\mathcal{P}_f$ using μ_e and μ_v
8	$\delta \leftarrow \min\{d'_f, \min_{e(c)\in P} \frac{\gamma}{A_{e(c)P}}, \min_{v\in V} \frac{\kappa(v)}{B_{vP}}\}$
9	$d'_f \leftarrow d'_f - \delta$
10	$x_f(P) \leftarrow x_f(P) + \delta$
11	$\forall e(c) \in E_c$ s.t. $A_{e(c)P} \neq 0$, $\mu_e \leftarrow \mu_e(1+\epsilon\delta A_{e(c)P}/\gamma)$
12	$\forall v \in V$ s.t. $B_{vP} \neq 0$, $\mu_v \leftarrow \mu_v(1+\epsilon\delta B_{vP}/\kappa(v))$
13	**end while**
14	**end for**
15	**end for**

Table VI.2: Routing Algorithm Under Fixed Demand

then filling traffic to this path by its bottleneck capacity (Lines 8 to 10). Then we update the prices for the interference sets and the nodes appeared in this path based on the function defined in Line 11 and Line 12. We keep filling traffic to flow f in the above fashion until all d_f units are routed. This procedure is repeated until the weighted aggregated price of the interference sets and the nodes exceeds 1 (Line 3).

We formally analyze the properties of our algorithm in the following theorem. The proofs of the theorems in this chapter are available in the Appendix II.

Theorem 1: If $\beta = ((|E_c| + |V|)/(1-\epsilon))^{-1/\epsilon}$, then the final flow generated by **FMR** is at least $(1-3\epsilon)$ times the optimal value of $\mathbf{P}$. The running time is $O(\frac{1}{\epsilon^2}[\log(|E_c|+|V|)(2|F|\log|F|+|E_c|+|V|)+\log U)]) \cdot T_{mp}$, where U is the length of the longest path in G, and T_{mp} is the running time to find the shortest path.

We have estimated the time complexity of the corresponding algorithm in the previous chapter in terms of $|V|$. Although the exact time complexity has changed due to the introduction of channels, the general upper bound in terms of $|V|$ remains $|V|^4 \log V$ assuming the channel number is a constant.

Uncertain Demand Mesh Network Routing

Now we proceed to investigate the throughput optimization routing problem for wireless mesh backbone network when the aggregated traffic demand is uncertain. We model such uncertain traffic demand of an aggregated flow $f \in F$ using a random variable D_f. We assume that D_f follows the following discrete probability distribution $Pr(D_f = d_f^i) = q_f^i$, where $\mathcal{D}_f = \{d_f^1, d_f^2, ..., d_f^m\}$ is the set of of values for D_f with non-zero probabilities. Let $\boldsymbol{d} = (d_f, d_f \in \mathcal{D}_f, f \in F)$ be a sample traffic demand vector, $\boldsymbol{D}$ be the corresponding random variable, and $\mathcal{D}$ be the sample space. Thus the distribution of $\boldsymbol{D}$ is given by the joint distribution of these random variables: $Pr(\boldsymbol{D} = \boldsymbol{d}) = Pr(D_f = d_f^i, f \in F)$.

Let us consider a traffic routing solution $(x_f(P), P \in \mathcal{P}_f, f \in F)$ that satisfies the capacity and node-radio constraints (Inequality (VI.5) and (VI.6)). It is obvious that λ is a function of $\boldsymbol{d}$: $\lambda(\boldsymbol{d}) = \min_{f \in F}\{\frac{x_f}{d_f}\}$, where $x_f = \sum_{P \in \mathcal{P}_f} x_f(P)$. Further let us consider the optimal routing solution under demand vector $\boldsymbol{d}$. Such a solution could be easily derived based on Algorithm I shown in Table VI.2. We denote the optimal value of λ as $\lambda^*(\boldsymbol{d})$. We further define the *performance ratio* ω of routing solution $(x_f(P), P \in \mathcal{P}_f, f \in F)$ as $\omega(\boldsymbol{d}) = \frac{\lambda(\boldsymbol{d})}{\lambda^*(\boldsymbol{d})}$

Obviously, the performance ratio is also a random variable under uncertain demand. We denote it as Ω which is a function of random variable $\boldsymbol{D}$. Now we extend the wireless mesh network routing problem to handle such uncertain demand. Our goal is to maximize the expected value of Ω, which is given by $E(\Omega) = Pr(\boldsymbol{D} = \boldsymbol{d}) \times \frac{\lambda(\boldsymbol{d})}{\lambda^*(\boldsymbol{d})}$

We abbreviate $Pr(\boldsymbol{D} = \boldsymbol{d})$ as $p(\boldsymbol{d})$. It is obvious that $\sum_{d \in \mathcal{D}} p(\boldsymbol{d}) = 1$. Formally, we formulate the throughput optimization routing problem for wireless mesh backbone network under uncertain traffic demand as follows.

$$\mathbf{P_U}:$$

$$\text{maximize} \quad \sum_{\boldsymbol{d} \in \mathcal{D}} p(\boldsymbol{d}) \frac{\lambda(\boldsymbol{d})}{\lambda^*(\boldsymbol{d})} \tag{VI.11}$$

$$\text{subject to} \quad \forall \boldsymbol{d} \in \mathcal{D}, \text{where } \boldsymbol{d} = (d_f, f \in F)$$

$$\sum_{P \in \mathcal{P}_f} x_f(P) \geq \lambda(\boldsymbol{d}) \cdot d_f, \forall f \in F \tag{VI.12}$$

$$\sum_{f \in F} \sum_{P \in \mathcal{P}_f} x_f(P) A_{e(c)P} \leq \gamma(\Delta), \forall e(c) \in E_c \tag{VI.13}$$

$$\sum_{f \in F} \sum_{P \in \mathcal{P}_f} x_f(P) B_{vP} \leq \kappa(v), \forall v \in V \tag{VI.14}$$

$$\lambda \geq 0, x_f(P) \geq 0, \forall f \in F, \forall P \in \mathcal{P}_f \tag{VI.15}$$

Similar to problem $\mathbf{P_T}$, the constraints of $\mathbf{P_U}$ come from the fairness requirement and the wireless mesh network capacity. In particular, Inequality (VI.12) enforces fairness for all demand $\boldsymbol{d} \in \mathcal{D}$, and Inequality (VI.13) enforces capacity constraint as Inequality (VI.5) in problem $\mathbf{P_T}$.

Now we consider the dual problem $\mathbf{D_U}$ of $\mathbf{P_U}$. Similar to $\mathbf{D_T}$, the objective of $\mathbf{D_U}$ is to minimize the aggregated price for all adjusted interference sets. However, in Inequality (VI.18), for each sample demand vector $\boldsymbol{d}$, the aggregated price of all flows weighted by their demand needs to be larger than its probability.

$$\mathbf{D_U}: \quad \text{minimize} \quad \sum_{e(c)\in E_c} \gamma(\Delta)\cdot\mu_e + \sum_{v\in V}\kappa(v)\mu_v \tag{VI.16}$$

$$\text{subject to} \quad \sum_{e(c)\in E_c} A_{e(c)P}\mu_e + \sum_{v\in V} B_{vP}\mu_v \geq \mu_f,$$

$$\forall f \in F, \forall P \in \mathcal{P}_f \tag{VI.17}$$

$$\sum_{f\in F}\mu_f d_f \geq \frac{p(\boldsymbol{d})}{\lambda^*(\boldsymbol{d})}, \forall \boldsymbol{d} \in \mathcal{D} \tag{VI.18}$$

$$\text{where } \boldsymbol{d} = (d_f, f \in F)$$

Now we present an approximation algorithm for $\mathbf{P_U}$ in Table VI.3. This algorithm (**UMR**) has the same initialization as the algorithm for problem $\mathbf{P_T}$ (**FMR**). Then we march into the iteration, in which we find $\boldsymbol{d}^{\min}$, the demand whose price $\mu^{\min}$ is the minimum among others (Lines 4 to 12). If $\mu^{\min} \geq 1$, then the algorithm stops (Lines 13 and 14), since Inequality (VI.17) and (VI.18) would be satisfied for all demand. Otherwise, we will increase the price of $\boldsymbol{d}^{\min}$ by routing more traffic through its node pairs. This procedure (Lines 16 to 23) is the same as what has been described in Lines 4 to 11 of **FMR** algorithm. Following the same proving sequence for **FMR**, we are able to prove the similar properties with **UMR**.

Theorem 2: If $\beta = ((|E_c| + |V|)/(1-\epsilon))^{-1/\epsilon}$, then the final flow generated by **UMR** is at least $(1-3\epsilon)$ times the optimal value of $\mathbf{P_U}$. The running time is $O(\frac{1}{\epsilon^2}[\log(|E_c| + |V|)(2|\mathcal{D}||F|\log|F| + |E_c| + |V|) + \log U)]) \cdot T_{mp}$, where U is the length of the longest path in G, T_{mp} is the running time to find the shortest path.

Similar to the previous algorithm, although the exact time complexity has changed due to the introduction of channels, the general upper bound in terms of $|V|$ remains $|V|^7$, which is the same as the one in the previous chapter, assuming the channel number is a constant.

UMR: Mesh Network Routing Under Uncertain Demand

$\forall e \in E$, $\gamma \leftarrow \gamma(\Delta)$, $\mu_e \leftarrow \beta/\gamma$, $\mu_v \leftarrow \beta/\kappa(v)$
$x_f(P) \leftarrow 0$, $\forall P \in \mathcal{P}_f$, $\forall f \in F$
loop
  for $\forall f \in F$ **do**
    $\bar{P} \leftarrow$ lowest priced path in $\mathcal{P}_f$ using μ_e, μ_v
    $\mu_f \leftarrow \sum_{e \in E} A_{e(c)\bar{P}} \mu_e + B_{v\bar{P}} \mu_v$
  end for
  for $\forall \boldsymbol{d} \in \mathcal{D}$ **do**
    $\mu_{\boldsymbol{d}} \leftarrow \sum_{f \in F} \mu_f d_f \frac{\lambda^*(\boldsymbol{d})}{p(\boldsymbol{d})}$
  end for
  $\mu^{\min} \leftarrow \min_{\boldsymbol{d} \in \mathcal{D}} \mu_{\boldsymbol{d}}$
  $\boldsymbol{d}^{\min} \leftarrow \arg\min_{\boldsymbol{d} \in \mathcal{D}} \mu^{\min}$
  if $\mu^{\min} \geq 1$
    return
  for $\forall f \in F$ **do**
    $d'_f \leftarrow d_f^{\min}$
    while $d'_f > 0$ **do**
      $P \leftarrow$ lowest priced path in $\mathcal{P}_f$ using μ_e, μ_v
      $\delta \leftarrow \min\{d'_f, \min_{e(c) \in P} \frac{\gamma}{A_{e(c)P}}, \min_{v \in V} \frac{\kappa(v)}{B_{vP}}\}$
      $d'_f \leftarrow d'_f - \delta$
      $x_f(P) \leftarrow x_f(P) + \delta$
      $\forall e(c) \in E_c$ s.t. $A_{e(c)P} \neq 0$, $\mu_e \leftarrow \mu_e(1+ \epsilon\delta A_{e(c)P}/\gamma)$
      $\forall v \in V$ s.t. $B_{vP} \neq 0$, $\mu_v \leftarrow \mu_v(1 + \epsilon\delta B_{vP}/\kappa(v))$
    end while
  end for
end loop

Table VI.3: Routing Algorithm Under Uncertain Demand

Channel Assignment Algorithms

The previous solutions provide a theoretical upper bound that a network can achieve with given constraints. However, the solutions may not be feasible in real network implementation because they do not include detailed information such as how channels are assigned or links are scheduled. It requires further algorithms to design feasible implementations based on the optimal solutions computed from LP. Those algorithms can provide detailed channel assignment and scheduling solutions for WMNs so that traffic can be routed following the solutions during network implementation phase. It is easy to see that a good channel assignment and scheduling algorithm should be able to achieve performance that is close to the theoretical upper bound.

Although it is not the focus of our work to design a channel assignment and scheduling algorithm to optimize the network performance, it makes the whole solution complete by adding this approach into our framework. Algorithms for wireless network channel assignment have been extensively discussed in several literatures [37, 17, 103, 38, 49, 104]. Instead of proposing a new algorithm for channel assignment, we utilize solutions from Kodialam et al.'s channel assignment algorithms [38] to provide further solutions to our routing algorithm results. A major reason for choosing this algorithm as part of our framework is that the results of our routing algorithms can be applied as the input of Kodialam et al.'s algorithms with little modification.

In Kodialam et al.'s work, they proposed two channel assignment algorithms. One is static channel assignment, where link channels are fixed at the beginning and won't change during the routing. The other is dynamic channel assignment, where links may switch to different channels at different time slot. The dynamic channel assignment has the most flexibility of channel assignment and better performance than static channel assignment, but frequent channel switch is an overhead to network operations. On the other hand, static channel assignment is a special case of the dynamic one. Fixed channel assignment may restrict exploring potential performance improvement in dynamic channel assignment, but the simplicity makes it easy to deploy and use.

Static Channel Assignment Algorithm

The static channel assignment algorithm assigns link channels at the beginning and the assignment remains unchanged for the rest of time slots. The main idea of static channel assignment algorithms is to balance traffic assigned to each interference set so that none of the interference sets are overloaded by any channel, otherwise a large amount of time slots will be used to solve conflict caused by heavy traffic assigned to a single interference set. Table VI.4 is the detailed description of the whole algorithm.

Static Channel Assignment Algorithm
$\mu_i = 0$, $\forall i \in$ interference sets I; $M = \phi$
$t(e) = \sum_{f\in F}\sum_{P\in\mathcal{P}_f \&\& P\ni e} x_f(P)$, $\forall e \in E$
while $\sum_{e\in E} t(e) > 0$
for $e \in E \backslash M$
$m(e,c) = \max_{i\ni(e,c)} \mu_i$, $\forall c \in C$
$w(e) = \min_c m(e,c)$
$b(e) = \arg\min_c m(e,c)$
end for
$\delta = \arg\min_{e\notin M} w(e)$
assign $w(\delta)$ to channel $b(\delta)$
$\mu_i = \mu_i + t(\delta)$, $\forall i \in I \&\& i \ni (l, b(\delta))$
$f_{b(\delta)}(\delta) = t(\delta)$; $f_c(\delta) = 0$, $\forall c \neq b(\delta)$
$M = M \cup l$; $t(\delta) = 0$
end while

Table VI.4: Static Channel Assignment Algorithm

In this algorithm, we assume that only one channel is assigned to each link for simplicity. For each pair of (e, c), e represents a link and c represents a channel. The definition of $m(e, c) = \max_{i \ni (e,c)} \mu_i$ tries to determine maximum load of each interference set containing the link pair (e, c). Then the algorithm chooses a link that has the minimum value of $\min_c m(e, c)$, assigns the channel to the corresponding c and re-allocate all traffic associated with link e to (e, c). The major reason of using min-max allocation in this algorithm is to balance the traffic load on interference sets.

The static channel assignment algorithm provides a result with flows are assigned to some specific channels. Based on this solution, Kodialam et al. use a greedy coloring algorithm to assign channel traffic to each time slot. The greedy scheduling algorithm first multiples all flows by a large number X and ignore the fractional part so that all flows are integer. Next, it chooses a link that has the highest residual traffic and assign links that are not interfered with to the smallest time-slot. Then it reduces all newly scheduled flow based on the time slot assigned, and repeats the whole process until all the flows have been scheduled.

If N denotes the maximum number of time slots taken by all the channels, then then new scaling factor λ' after scheduling is calculated as

$$\lambda' = \frac{X}{N} \times \lambda \tag{VI.19}$$

where λ is the scaling factor computed in the optimal routing in the first phase.

Dynamic Channel Assignment Algorithm

The static channel assignment algorithm assigns link channels at the beginning of time slot and the channel assignment remains the same after the rest of time period. Dynamic channel assignment algorithm, on the other hand, allows links to switch different channel every T time slots, which implies that channel assignment needs to be reschedule and coordinated at the end of every T time slots. When T is equal to 1, the algorithm has the most flexibility to achieve optimal performance. However, frequent coordination can be an overhead to networks. The static channel assignment algorithm can be treated as a special version of the dynamic channel assignment algorithm when T becomes infinite large.

The main idea of Kodialam et al.'s dynamic channel assignment algorithm is to pack flow greedily in each time slot. Similar to the static channel assignment algorithm, it first packs solutions from LP for optimal routing so that traffic from the same link but different channels is merged into a single link. Then it picks the link that has the highest remaining traffic and assigns the link to the channel that has the highest remaining capacity. Next, it checks links with remaining traffic in descend order and assign links with channels that have the highest remaining capacity. The remaining links that are not assigned with any channel during that time slot are moved to next time slot. The algorithm exits when all the traffic is assigned to certain channels. Table VI.5 is the detailed description of the whole algorithm.

The new scaling factor λ' after scheduling is calculated also as

$$\lambda' = \frac{X}{N} \times \lambda \tag{VI.20}$$

Dynamic Channel Assignment Algorithm

$N = 0$; $M = E$
$t(e) = \sum_{f \in F} \sum_{P \in \mathcal{P}_f \&\& P \ni e} x_f(P)$, $\forall e \in E$
while $M \neq \phi$
  $N = N + T$
  Sort E in descending order of $t(e)$
  $y_i = 0$, $\forall i \in$ interference sets I
  for each e in sorted E
    if $\exists l$ such that $y_i == 0$ $\forall i \in I \&\& i \ni (e, l)$
      $l = \arg\max_c \mathcal{C}_c(e)$
      Assign e to channel l
      $t(e) = t(e) - T \cdot \mathcal{C}_c(e)$
      if $t(e) == 0$ **then** $M = M \backslash e$
      $y_i = 1$, $\forall i \in I \&\& i \ni (e, l)$
    end if
  end for
end while

Table VI.5: Dynamic Channel Assignment Algorithm

where λ is the scaling factor computed in the optimal routing in the first phase.

Simulation Study

We evaluate the performance of our algorithms via a simulation study. In the simulated wireless mesh network, 60 mesh nodes are randomly deployed over a $1000 \times 2000m^2$ region. 20 nodes at the edge of this network are selected as the local access points (LAP) that forward traffic for clients. 4 nodes in the center of the deploy region are selected as the gateway access points. The simulated network topology is shown in Fig. VI.1. Each mesh node has a transmission range of $250m$ and an interference range of $500m$, which means $\Delta = 2$. The channel capacity $\phi_c(e)$ is the same for all links e and channels c, which is set as 54 Mbps. In the basic setting, each mesh nodes are equipped with 3 radio interfaces. And there are 3 orthogonal channels in the network. Aside from this basic setting, we have also evaluated the performance of our algorithms with different configurations of radio and channel numbers, which we will show in the later part of this section.

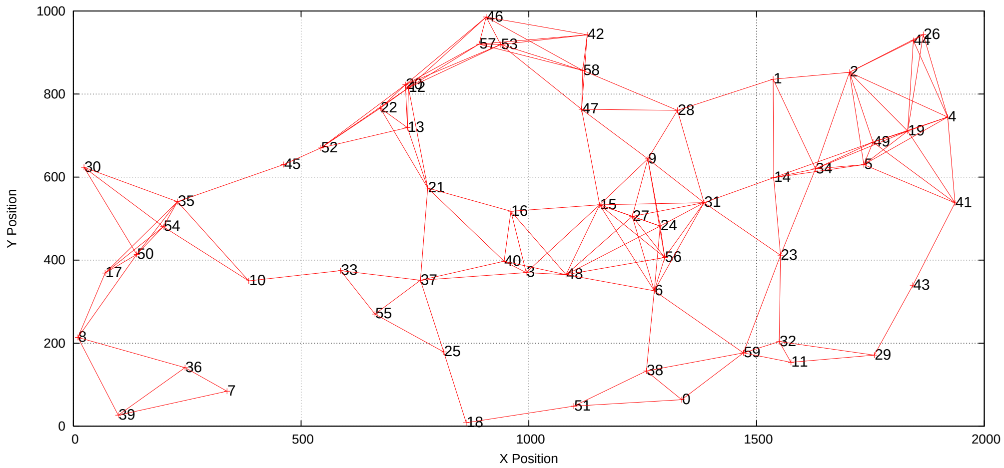


Figure VI.1: Mesh Network Topology.

To realistically simulate the traffic demand at each LAP, we employ the traces collected in the campus wireless LAN network. The network traces used in this work are collected in Spring 2002 at Dartmouth College and provided by CRAW-DAD [100]. By analyzing the *snmp* log trace at each access point, we are able to derive its 1108-hour incoming and outgoing traffic volume since 12:00AM, March 25, 2002 EST. We select the access points from the Dartmouth campus wireless LAN and assign their traffic traces to the LAPs in our simulation. The traffic assignment is given in Table VI.6.

AP	31AP3	34AP5	55AP4	57AP2	62AP3	62AP4	82AP4	94AP1	94AP3	94AP8
Node ID	22	18	57	5	55	20	53	3	56	27
AP	27AP3	3AP3	21AP2	23AP4	33AP2	62AP2	82AP3	84AP1	90AP2	97AP2
Node ID	9	23	25	33	19	35	58	42	6	48

Table VI.6: Overview of Traffic Demand Assignment

We evaluate and compare different traffic prediction and routing strategies for this simulated network. In particular, we consider the following strategies.

- *Oracle Routing (OR).* In this strategy, the traffic demand is known a priori. It runs the **FMR** algorithm (presented in Tab. VI.2) based on this demand. This solution runs every hour based on the up-to-date traffic demand from the trace and returns the optimal set of routes. This ideal strategy is designed to return the benchmark result, which the rest of the practical strategies compare to.

- *Mean-Value Prediction Routing (MVPR).* This strategy does not know the traffic demand a priori. Instead, it only predicts the traffic demand based on its historical data. In particular, it employs the mean value prediction model and runs the **FMR** algorithm based on this predicted demand. This solution also runs every hour to provide the set of routes for the next hour.

- *Statistical-Distribution Prediction Routing (SDPR).* Similar to MVPR, this strategy also relies on traffic prediction. It predicts not only the mean-value of the traffic demand in the next hour, but also its distribution. It runs the **UMR** algorithm (presented in Tab. VI.3) with the predicted traffic demand distribution as its input. Since **UMR** only accepts discrete probability distribution, we need to discretize the demand distribution by sampling the following values, the mean value μ, and values $\mu - \sigma$, $\mu + \sigma$, $\mu - 2\sigma$, and $\mu - 2\sigma$. Since about 95% of all traffic demand values fall within the range $[\mu - 2\sigma, \mu + 2\sigma]$, we ignore the values which has a probability smaller than 5%.

- *Shortest-Path Routing (SPR).* This strategy is agnostic of traffic demand, and returns fixed routing solution purely based on the shortest distance (number of hops) from each mesh node to the gateway. The purpose to evaluate this strategy is to quantitatively contrast the advantage of our traffic-predictive routing strategies.

Note that the flows derived from the above routing strategies will be adjusted by the channel assignment, post processing and flow scaling algorithms in [18]. We denote the final rate of flow f along path P as $x_f^A = \sum_{P \in \mathcal{P}_f} x_f^A(P)$. This is the maximum flow throughput under the fairness constraint weighted by the traffic demand, which maximizes the scaling factor λ. However, for performance study, λ is not a suitable performance metric. First, we are more interested in the network performance (*i.e.*, congestion) incurred by the given traffic demand, instead of the achievable throughput. Second, the absolute value of λ could be misleading, especially when the actual demand is not the same as the predicted demand which is being used for routing.

Now we proceed to define the performance metric we use in the simulation study. First, we scale the achievable flow rate x_f^A derived from the routing and channel assignment process by its actual traffic demand d_f:

$$x'_f(P) = x_f^A(P) \cdot \frac{d_f}{x_f^A} \tag{VI.21}$$

$x'_f(P)$ is the actual traffic load that is imposed on path P under our routing and channel assignment scheme. Thus the traffic being routed within the interference set $I_e(c)$ over channel c is given by $\sum_{f \in F} \sum_{P \in \mathcal{P}_f} x'_f(P) A_{e(c)P}$. We define the *congestion of an interference set* $I_e(c)$ using its utilization and denote it as

$\theta_e^{ch}(c) = \frac{\sum_{f \in F} \sum_{P \in \mathcal{P}_f} x'_f(P) A_{e(c)P}}{\gamma(\Delta)}$. Then $\theta^{ch} = \max_{e(c) \in E_c} \theta_e^{ch}(c)$ is the maximum congestion among all the interference sets. We further consider the congestion at a single mesh node incurred by the traffic from all channels. The *congestion of a node* v is defined as $\theta_v^{rd} = \frac{\sum_{f \in F} \sum_{P \in \mathcal{P}_f} x'_f(P) B_{vP}}{\kappa(v)}$. And $\theta^{rd} = \max_{v \in V} \theta_v^{rd}$. Finally, the *network congestion* θ is defined as $\theta = \max\{\theta^{rd}, \theta^{ch}\}$.

Finally, we should point out that our algorithms are centralized algorithms which collect global information before the computation. That means, traffic will be automatically rerouted once the centralized node notice network change, including not only traffic but also node failure and/or new node joining. The sensitiveness to the topology change depends on how quickly the centralized node receive the topology change.

Simulation Results

We experiment with the above routing strategies along the time range $[108, 1108]$, a 1000-hour period excerpted from the trace[1]. Note that all the simulation results presented in this section are using 108 as the zero point.

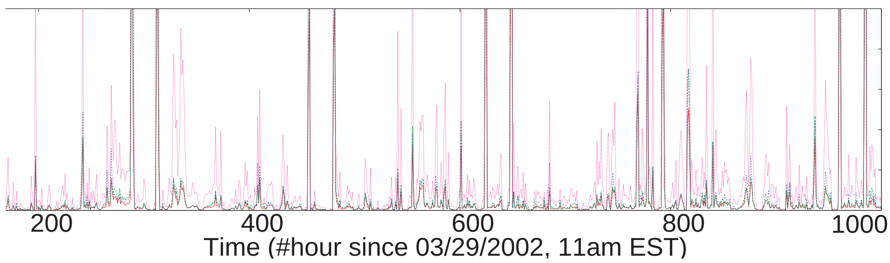
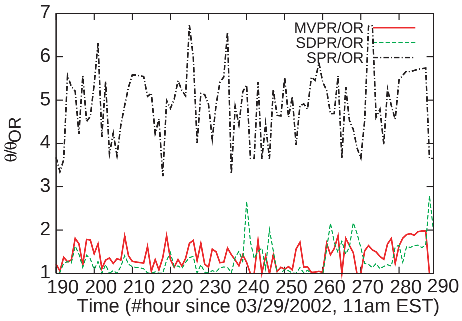


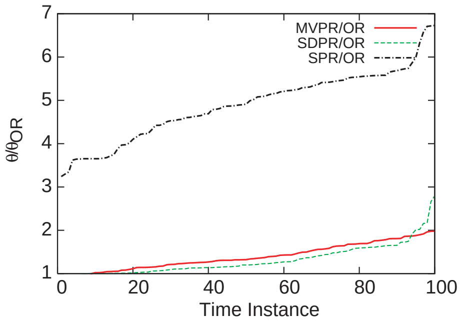
Figure VI.2: Overview of All Strategies

[1]Note that the beginning part of the trace $[0, 107]$ is used as training data, thus is not included in the simulation result.

We start by presenting the congestion achieved by all strategies (*OR*, *MVPR*, *SDPR*, and *SPR*) during the entire 1000-hour simulation period. As seen in Fig. VI.2, *OR* constantly achieves the minimum worst-case congestion among others, due to its unrealistic capability to know the actual traffic demand. We note that the burstiness of θ applies to all strategies including *OR*. Such observation comes from the burstiness of the traffic load in the *snmp* log trace, which is caused by the insufficient level of traffic multiplexing at wireless local access points.



(a) Congestion Ratio ($\frac{\theta}{\theta_{OR}}$)



(b) Sorted View

Figure VI.3: Comparison to OR

To filter out the noise caused by traffic burstiness, in Fig. VI.3(a), we normalize θ achieved by other strategies by the same value of *OR*. Since *OR* always achieves the minimum θ among others, this ratio will end up at least 1. Also we take a close-up look during the hour range $[190, 290]$. Here, the *MVPR* and *SDPR* strategies achieve less than 2 times of the optimal congestion in most cases, while the *SPR* strategy can only achieve $4-7$ times of the optimal performance. The above observations get clearer when we sort out the normalized congestion ratio for the three strategies in Fig. VI.3(b). It is clear that our *MVPR* and *SDPR* strategies, which integrate the traffic prediction with the optimal routing, outperform the *SPR* strategy which is agnostic about the traffic demand. Further, *SDPR* achieves lower congestion than *MVPR* in most of the time due to more comprehensive representation of the traffic demand estimation. However, in a few cases (less than 10% of the time), the worst-case congestion of *SDPR* is higher than *MVPR*. This problem can be mostly attributed to the inaccuracy of traffic prediction.

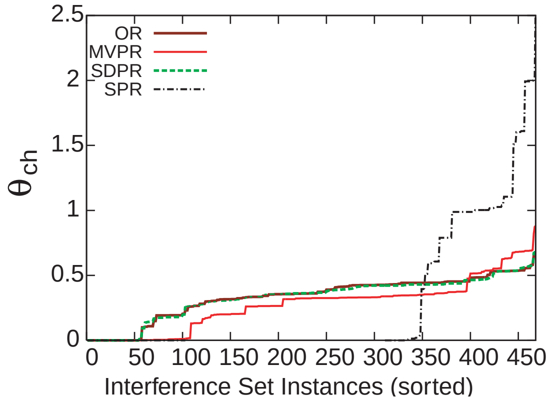
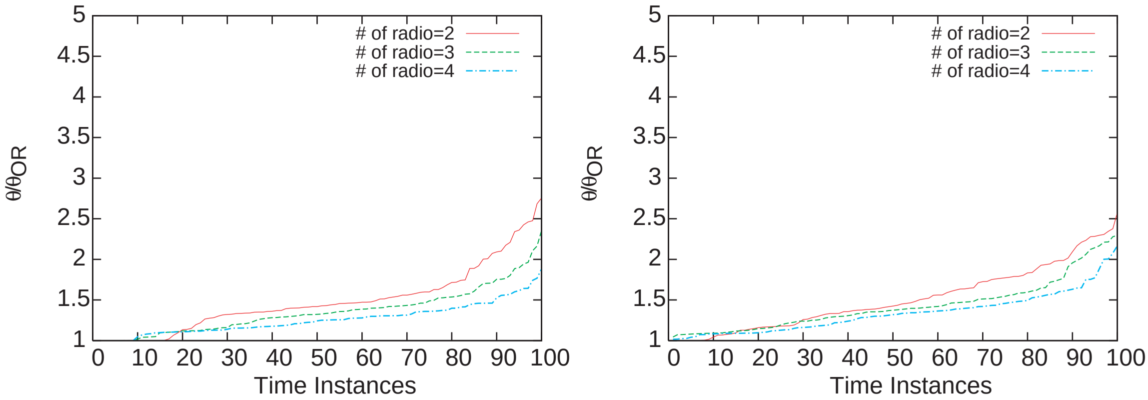


Figure VI.4: Adjusted Interference Set Sorted By Congestion

Next, we take a closer look at each strategy's ability to balance the traffic within the mesh network. In Fig. VI.4, we unfold a single time instance at hour 271 and exhibit the congestion $\theta^{ch}_{e(c)}$ at each interference set $I_e(c)$ resulted from each strategy. In order to achieve the lowest worst-case congestion, a good strategy should maximally even out the traffic routed through all interference sets. Obviously, *OR* achieves such a balance, which resulted in the best θ value 0.65. *SPR* has the highest θ value as more than 2. The results for *MVPR* and *SDPR* are 0.8 and 0.7 respectively. We also observe that the distribution of $\theta^{ch}_{e(c)}$ under the *SDPR* strategy closely matches the *OR* strategy.

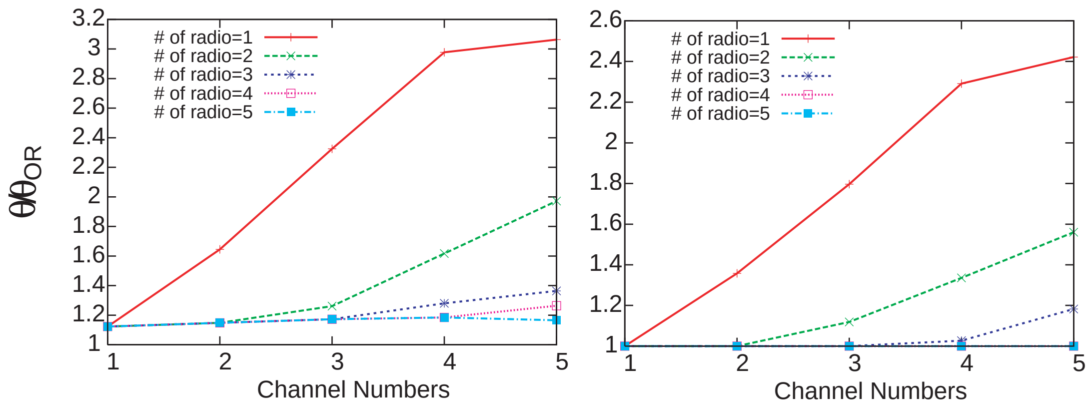
In what follows, we alter our simulation configurations to examine the abilities of different strategies at adapting various network settings, such as radio interface numbers and channel numbers. Here, we focus on the traffic prediction strategies,



(a) MVPR (b) SDPR

Figure VI.5: Impact of Number of Radio Interfaces

namely, *MVPR* and *SDPR*. Also we plot their performances by the congestion ratio θ/θ_{OR} normalized by the *OR* routing results. We first vary the number of radio interfaces from 2 to 4 and study the congestion θ during the time interval $[190, 290]$. Fig. VI.5 plots the sorted normalized congestion $\frac{\theta}{\theta_{OR}}$ of the two strategies. Comparing these two figures, we could see that the *SDPR* strategy performs slightly better than the *MVPR* strategy. The improvement of both strategies over the *OR* strategy increases (*i.e.*, normalized congestion decreases) with the radio number.



(a) MVPR (b) SDPR

Figure VI.6: Impact of Channel/Radio

Also, Fig. VI.6 plots the normalized congestion under different radio and channel numbers at a single time instance 271 for these two strategies. The results show that the improvement of both strategies over the *OR* strategy decreases with the channel number. This is because when the network has more channels, the algorithms are likely to find more paths and the prediction error is more likely to be magnified.

As we point out in earlier sections, *OR*, *MVPR*, and *SDPR* only provide a theoretical upper bond for each scenario. It requires actual channel assignment together with scheduling algorithm to deliver a feasible solution. The implemented algorithm may affect computed performance at difference levels.

In our research, we are interested in knowing how those algorithms can impact the original performance of our predicted algorithm. We picked a period of time from our simulation, and implemented *SDPR* results using both static and dynamic channel assignment algorithms. Fig. VI.7 shows throughput performance of static and dynamic channel assignment algorithms comparing to original *SDPR* throughput. It is easy to notice that dynamic channel assignment can produce thoughput that is almost the same as the original *SDPR*. Static channel assignment algorithm, on the other hand, can only generate about half of original *SDPR* throughput on average. The reason for this difference is that the dynamic feature of channel assignment gives maximum flexibility and allows links to reuse used channels as much as possible so that maximum traffic can be routed over links.

Finally, channel assignment and scheduling algorithms can ease traffic congestion over networks, which is also supported by Fig. VI.8. It compares the traffic congestion over the same period to *SDPR*. Both static and dynamic channel assignment algorithms show that the network become less congested after applying those algorithms overall. However, comparing to dynamic channel assignment,

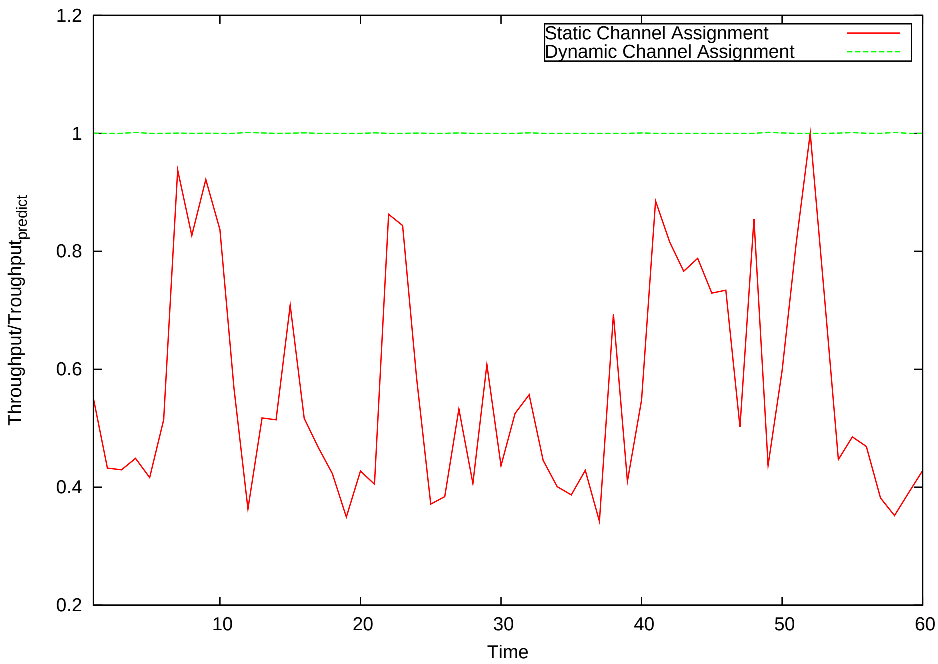


Figure VI.7: Throughput after SDPR is implemented by channel assignment algorithms

static channel assignment does not reduce traffic congestion heavily. In some time slots, the performance is even worse then *SDPR* algorithm. It is mainly because that static channel assignment merges traffic from different channels of the same link, and it actually has worse congestion than *SDPR* algorithm. However, the reduced throughput of static channel algorithm reduce the congestion. So overall, it does a poor job in reducing congestion. Dynamic channel assignment algorithm, on the other hand, distributed traffic to different channels and make the network less congested.

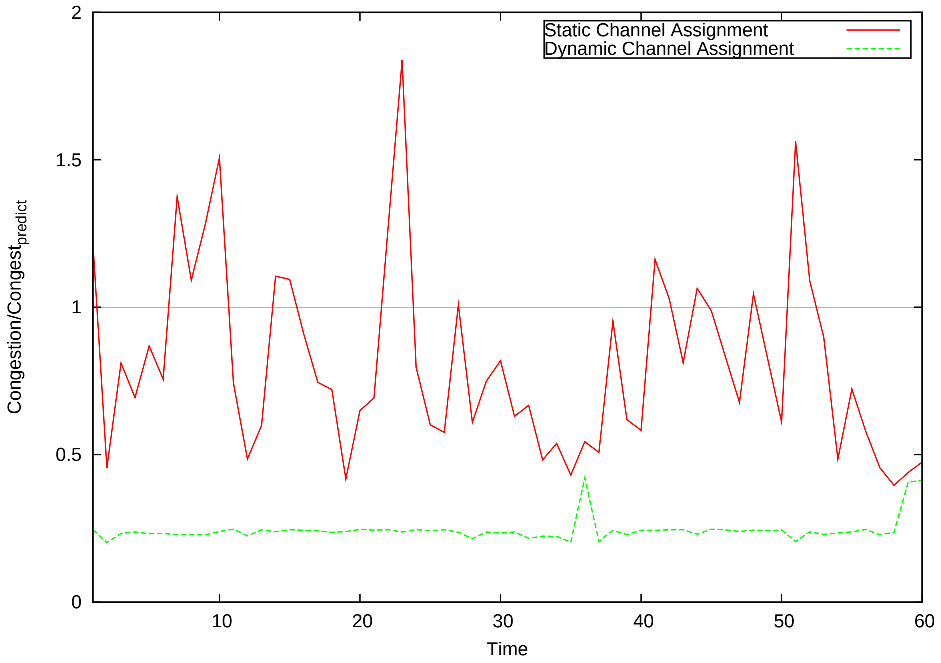


Figure VI.8: Congestion after SDPR is implemented by channel assignment algorithms

CHAPTER VII

ALGORITHM EXTENSION FOR LOSSY WIRELESS MESH NETWORKS

Solution Overview

As an important factor in wireless mesh network routing, wireless link quality, especially link loss, has a direct impact on traffic routing. Routing algorithms without considering link loss may heavily use links that have high loss ratio and can lead to poor network performance. Link loss aware routing algorithms can accurately reflect physical network environment and design routing paths by adapting this feature. In this chapter, we extend the routing joint channel assignment and routing algorithms for stable wireless mesh networks introduced in the previous chapter. We incorporate the lossy link characteristic into the modeling and penalize links with higher loss ratio so that routing paths will favor links with lower loss ratio.

The joint channel assignment and routing algorithms for lossy WMNs introduced in this chapter are the extension of solutions for stable WMNs in Chapter VI, so the solution may look similar to each other. However, the solutions after introducing the link loss concept into the formulation are adaptive to lossy network environment.

We assume that the system operates synchronously in a time-slotted mode. The result we obtain will provide an upper bound for systems using IEEE 802.11 MAC. We further assume that the traffic between a local access point and the Internet could be infinitesimally divided and routed over multiple paths to multiple gateways achieving the optimal load balancing and the least congestion.

To study the optimal routing problem, we also need to understand the link stability problem in wireless links. Links in wireless networks are less stable compared to traditional wireline network links, and packets in wireless transmission can be lost due to various reasons such as wireless noise. One common metric for describing wireless link transmission status is link loss ratio, which measures the percentage of packet lost over a link during a certain time period. For ease of exposition, we use delivery ratio $\gamma_e(c)$, an opposite metric of link loss ratio, to describe the ratio of packets received over link $e(c)$. Based on this metric, a node will have a receiving rate $y_e^r(c)$ equal to $\gamma_e(c)y_e^s(c)$ if the other node of a link has a sending rate of $y_e^s(c)$. Further, we can have *effective link capacity* $\phi_e'(c) = \gamma_e(c)\phi_e(c)$, which represents the actual maximum traffic that can be transmitted through link $e(c)$ compared to the claimed link capacity $\phi_e(c)$.

Our algorithms are flow based algorithms, while traffic in WMNs is transmitted packet by packet. A flow-based solution may not be schedulable at the packet level. Our algorithms should guarantee that the solution is not only feasible at the theoretical level, but also practical at the implementation level. It is important to understand the constraint of the flow rates in order to address the optimal routing problem.

We still use Claim 2 introduced in Chapter III as the necessary condition for the following formulation. However, the definitions of some variables in Claim 2 have been refined in the new context of lossy wireless networks. Let $\boldsymbol{y} = (y_e^s(c), e \in E)$ denote the wireless link rate vector, where $y_e^s(c)$ is the aggregated flow sending rate along wireless link $e(c)$. Link rate vector $\boldsymbol{y}$ is said to be schedulable, if there exists a stable schedule that ensures every packet transmission with a bounded delay. Essentially, the constraint of the flow rates is defined by the schedulable region of the link rate vector $\boldsymbol{y}$. Let $\rho_e(c) = \frac{y_e^s(c)}{\phi_e(c)} = \frac{y_e^r(c)}{\phi'_e(c)}$ be the utilization of channel c over link e, and $E(v)$ be the set of links that is adjacent to node v.

The focus of this chapter is to investigate the optimal routing scheme under dynamic traffic based on the above necessary conditions of channel assignment and schedulability. Once the flow routes are derived, we simply apply the same method presented in [18] to adjust the flow routes and scale the flow rates to yield a feasible routing and channel assignment.

The objective is to determine the necessary and sufficient conditions for the link flow rates to be achievable in the network in terms of a valid schedule. We define a $0-1$ scheduling variable $Y_e(c)$

$$Y_e^t(c) = \begin{cases} 1 & \text{if link } e \text{ is active on channel } c \text{ in time slot } t \\ 0 & \text{otherwise} \end{cases}$$

The notations used in this chapter are summarized in Table VII.1.

Fixed Demand Multi-Radio Multi-Channel Mesh Network Routing (FM^3R)

We first study the formulation of throughput optimization routing problem in a wireless mesh backbone network under the fixed traffic demand. We use d_f to denote the demand of flow f and $\boldsymbol{d} = (d_f, f \in F)$ to denote the demand vector consisting of all flow demands. Consider the fairness constraint that, for each flow f, its throughput being routed is in proportion to its demand d_f. Our goal is to maximize λ (so called *scaling factor*) where at least $\lambda \cdot d_f$ amount of throughput can be routed for flow f.

Notation	Definition
$G = (V, E)$	Network
$G' = (V', E')$	Network with virtual gateway/links
$u \in V$	Node
$e = (u, v) \in E$	Edge connecting nodes u and v
$f \in F$	Aggregated flow
$\boldsymbol{x} = (x_f, f \in F)$	Aggregated flow rate vector
$\boldsymbol{y} = (y_e, e \in E)$	Wireless link rate vector
$\boldsymbol{d} = (d_f, f \in F)$	Flow traffic demands
$p(\boldsymbol{d})$	Probability of $\boldsymbol{d}$
$\mathcal{P}_f$	Set of paths that can route f
$x_f(P)$	Rate of flow f over path $P \in \mathcal{P}_f$
$\gamma_e(c)$	Effective receiving ratio of link $e(c)$
$\phi_e(c)$	Claimed link capacity of e on channel c
$\phi'_e(c)$	Effective link capacity of e on channel c
I_e	Interference set of $e \in E$
$A_{eP} = \|I_e \cap P\|$	Number of wireless links P passes in I_e
$\lambda = \min_{f \in F}\{\frac{x_f}{d_f}\}$	Scaling factor
μ_e	Price of I_e

Table VII.1: Notations

We assume an infinitesimally divisible flow model where the aggregated traffic flow could be routed over multiple paths and use $\mathcal{P}_f$ to denote the set of unicast paths that connect the source of f and w^*. Let $x_f(P)$ be the rate of flow f over path $P \in \mathcal{P}_f$. We define $x_f(P)$ as the receiving rate at the destination node of the flow $f(P)$ and have $x_f(P) \leq y_e^r(c), \forall e(c) \in P$. According to this definition, it is easy to see that in a lossy environment, a relaying node of a flow does not have equal incoming (receiving) flow rate and out (sending) flow rate because the sending flow will retransmit the lost packets. In the following description, we assume the rate is the receiving rate by default and only use the concept of sending rate whenever necessary. Obviously the link flow rate $y_e(c)$ is given by $y_e(c) = \sum_{f:P \in \mathcal{P}_f \& e(c) \in P} x_f(P)$, which is the sum of the flow rates that are routed through paths P passing edge $e(c) \in E_c$. Based on the necessary conditions of scheduling and channel assignment in Claim 2 (Chapter III)), we have that

$$\sum_{e'(c) \in I_e(c)} \frac{1}{\phi'_{e'(c)}} \sum_{f: P \in \mathcal{P}_f \& e'(c) \in P} x_f(P) \leq 1; \forall e(c) \in E_c \tag{VII.1}$$

$$\sum_{c \in C} \sum_{e'(c) \in E(v)} \frac{1}{\phi'_{e'(c)}} \sum_{f: P \in \mathcal{P}_f \& e'(c) \in P} x_f(P) \leq \kappa(v); \forall v \in V \tag{VII.2}$$

To simplify the above equations, we define $A_{e(c)P} = \sum_{e'(c) \in I_{e(c)}, e'(c) \in P} \frac{1}{\phi'_{e'(c)}}$ and $B_{vP} = \sum_{c \in C} \sum_{e'(c) \in E(v), e'(c) \in P} \frac{1}{\phi'_{e'(c)}}$. The throughput optimization routing with fairness constraint is then formulated as the following linear programming (LP) problem:

$$\mathbf{P_T}: \quad \text{maximize} \quad \lambda \tag{VII.3}$$

$$\text{subject to} \quad \sum_{P \in \mathcal{P}_f} x_f(P) \geq \lambda \cdot d_f, \forall f \in F \tag{VII.4}$$

$$\sum_{f \in F} \sum_{P \in \mathcal{P}_f} x_f(P) A_{e(c)P} \leq 1, \\ \forall e(c) \in E_c \tag{VII.5}$$

$$\sum_{f \in F} \sum_{P \in \mathcal{P}_f} x_f(P) B_{vP} \leq \kappa(v), \forall v \in V \tag{VII.6}$$

$$\lambda \geq 0, x_f(P) \geq 0, \forall f \in F, \forall P \in \mathcal{P}_f \tag{VII.7}$$

In this problem, the optimization objective is to maximize λ, such that at least $\lambda \cdot d_f$ units of data can be routed for each aggregated flow f with demand d_f. Inequality (VII.4) enforces fairness by requiring that the comparative ratio of traffic routed for different flows satisfies the comparative ratio of their demands. Inequality (VII.5) and (VII.6) come from the necessary conditions of channel assignment and scheduling. This problem formulation follows the same form as the maximum concurrent flow problem.

Problem $\mathbf{P_T}$ could be solved by a LP-solver such as [102]. To reduce the complexity for practical use, we present a fully polynomial time approximation algorithm for problem $\mathbf{P_T}$, which finds an ϵ-approximate solution. The key to a fast approximation algorithm lies on the dual of this problem, which is formulated as follows. We assign a price μ_e to each set $I_e(c)$ for $e(c) \in E_c$ and a price μ_v to each node $v \in V$. The objective is to minimize the aggregated price for all interference sets and all nodes. As the constraint, Inequality (VII.9) requires that the price $\sum_{e(c)\in E_c} A_{e(c)P}\mu_e + \sum_{v\in V} B_{vP}\mu_v$ of any path $P \in \mathcal{P}_f$ for flow f must be at least μ_f, the price of flow f. Further, Inequality (VII.10) requires that the weighted flow price μ_f over its demand d_f must be at least 1.

$$\mathbf{D_T}: \quad \text{minimize} \quad \sum_{e(c)\in E_c} \mu_e + \sum_{v\in V} \kappa(v)\mu_v \tag{VII.8}$$

$$\text{subject to} \quad \sum_{e(c)\in E_c} A_{e(c)P}\mu_e + \sum_{v\in V} B_{vP}\mu_v \geq \mu_f, \quad \forall f \in F, \forall P \in \mathcal{P}_f \tag{VII.9}$$

$$\sum_{f\in F} \mu_f d_f \geq 1 \tag{VII.10}$$

Based on the above dual problem $\mathbf{D_T}$, our fast approximation algorithm is presented in Table VII.2. The algorithm design follows the idea of [59]. In particular, Line 1 and Line 2 initialize the algorithm. Then for each flow f, we route d_f units of data. We do so by finding the lowest priced path in the path set $\mathcal{P}_f$ (Line 7),

FM^3R: **Mesh Network Routing Under Fixed Demand**	
1	$\forall e \in E$, $\mu_e \leftarrow \beta$, $\mu_v \leftarrow \beta/\kappa(v)$
2	$x_f(P) \leftarrow 0$, $\forall P \in \mathcal{P}_f$, $\forall f \in F$
3	**while** $\sum_{e(c)\in E(c)} \mu_e + \sum_{v\in V} \kappa(v)\mu_v < 1$
4	**for** $\forall f \in F$ **do**
5	$d'_f \leftarrow d_f$
6	**while** $\sum_{e(c)\in E(c)} \mu_e + \sum_{v\in V} \kappa(v)\mu_v < 1$ **and** $d'_f > 0$ **do**
7	$P \leftarrow$ lowest priced path in $\mathcal{P}_f$ using μ_e and μ_v
8	$\delta \leftarrow \min\{d'_f, \min_{e(c)\in P} \frac{1}{A_{e(c)P}}, \min_{v\in V} \frac{\kappa(v)}{B_{vP}}\}$
9	$d'_f \leftarrow d'_f - \delta$
10	$x_f(P) \leftarrow x_f(P) + \delta$
11	$\forall e(c) \in E_c$ s.t. $A_{e(c)P} \neq 0$, $\mu_e \leftarrow \mu_e(1 + \epsilon\delta A_{e(c)P})$
12	$\forall v \in V$ s.t. $B_{vP} \neq 0$, $\mu_v \leftarrow \mu_v(1 + \epsilon\delta B_{vP}/\kappa(v))$
13	**end while**
14	**end for**
15	**end for**

Table VII.2: Routing Algorithm Under Fixed Demand

then filling traffic to this path by its bottleneck capacity (Lines 8 to 10). Then we update the prices for the interference sets and the nodes appeared in this path based on the function defined in Line 11 and Line 12. We keep filling traffic to flow f in the above fashion until all d_f units are routed. This procedure is repeated until the weighted aggregated price of the interference sets and the nodes exceeds 1 (Line 3).

Uncertain Demand Multi-Radio Multi-Channel Mesh Network Routing (UM^3R)

Now we proceed to investigate the throughput optimization routing problem for wireless mesh backbone network when the aggregated traffic demand is uncertain. We model such uncertain traffic demand of an aggregated flow $f \in F$ using a random variable D_f. We assume that D_f follows the following discrete probability distribution $Pr(D_f = d_f^i) = q_f^i$, where $\mathcal{D}_f = \{d_f^1, d_f^2, ..., d_f^m\}$ is the set of of values for D_f with non-zero probabilities. Let $\boldsymbol{d} = (d_f, d_f \in \mathcal{D}_f, f \in F)$ be a sample traffic demand vector, $\boldsymbol{D}$ be the corresponding random variable, and $\mathcal{D}$ be the sample space. Thus the distribution of $\boldsymbol{D}$ is given by the joint distribution of these random variables: $Pr(\boldsymbol{D} = \boldsymbol{d}) = Pr(D_f = d_f^i, f \in F)$.

Let us consider a traffic routing solution $(x_f(P), P \in \mathcal{P}_f, f \in F)$ that satisfies the capacity and node-radio constraints (Inequality (VII.5) and (VII.6)). It is obvious that λ is a function of $\boldsymbol{d}$: $\lambda(\boldsymbol{d}) = \min_{f \in F}\{\frac{x_f}{d_f}\}$, where $x_f = \sum_{P \in \mathcal{P}_f} x_f(P)$. Further let us consider the optimal routing solution under demand vector $\boldsymbol{d}$. Such a solution could be easily derived based on Algorithm I shown in Table VII.2. We denote the optimal value of λ as $\lambda^*(\boldsymbol{d})$. We further define the *performance ratio* ω of routing solution $(x_f(P), P \in \mathcal{P}_f, f \in F)$ as $\omega(\boldsymbol{d}) = \frac{\lambda(\boldsymbol{d})}{\lambda^*(\boldsymbol{d})}$

Obviously, the performance ratio is also a random variable under uncertain demand. We denote it as Ω, which is a function of random variable $\boldsymbol{D}$. Now we extend the wireless mesh network routing problem to handle such uncertain demand. Our goal is to maximize the expected value of Ω, which is given by $E(\Omega) = Pr(\boldsymbol{D} = \boldsymbol{d}) \times \frac{\lambda(\boldsymbol{d})}{\lambda^*(\boldsymbol{d})}$

We abbreviate $Pr(\boldsymbol{D} = \boldsymbol{d})$ as $p(\boldsymbol{d})$. It is obvious that $\sum_{d \in \mathcal{D}} p(\boldsymbol{d}) = 1$. Formally, we formulate the throughput optimization routing problem for wireless mesh backbone network under uncertain traffic demand as follows.

$$
\begin{aligned}
\mathbf{P_U}: \text{maximize} \quad & \sum_{\boldsymbol{d} \in \mathcal{D}} p(\boldsymbol{d}) \frac{\lambda(\boldsymbol{d})}{\lambda^*(\boldsymbol{d})} && \text{(VII.11)} \\
\text{subject to} \quad & \forall \boldsymbol{d} \in \mathcal{D}, \text{where } \boldsymbol{d} = (d_f, f \in F) \\
& \sum_{P \in \mathcal{P}_f} x_f(P) \geq \lambda(\boldsymbol{d}) \cdot d_f, \forall f \in F && \text{(VII.12)} \\
& \sum_{f \in F} \sum_{P \in \mathcal{P}_f} x_f(P) A_{e(c)P} \leq 1, \forall e(c) \in E_c && \text{(VII.13)} \\
& \sum_{f \in F} \sum_{P \in \mathcal{P}_f} x_f(P) B_{vP} \leq \kappa(v), \forall v \in V && \text{(VII.14)} \\
& \lambda \geq 0, x_f(P) \geq 0, \forall f \in F, \forall P \in \mathcal{P}_f && \text{(VII.15)}
\end{aligned}
$$

Similar to problem $\mathbf{P_T}$, the constraints of $\mathbf{P_U}$ come from the fairness requirement and the wireless mesh network capacity. In particular, Inequality (VII.12) enforces fairness for all demand $\boldsymbol{d} \in \mathcal{D}$, and Inequality (VII.13) enforces capacity constraint as Inequality (VII.5) in problem $\mathbf{P_T}$.

Now we consider the dual problem $\mathbf{D_U}$ of $\mathbf{P_U}$. Similar to $\mathbf{D_T}$, the objective of $\mathbf{D_U}$ is to minimize the aggregated price for all adjusted interference sets. However, in Inequality (VII.18), for each sample demand vector $\boldsymbol{d}$, the aggregated price of all flows weighted by their demand needs to be larger than its probability.

$$\mathbf{D_U}: \quad \text{minimize} \quad \sum_{e(c)\in E_c} \mu_e + \sum_{v\in V} \kappa(v)\mu_v \tag{VII.16}$$

$$\text{subject to} \quad \sum_{e(c)\in E_c} A_{e(c)P}\mu_e + \sum_{v\in V} B_{vP}\mu_v \geq \mu_f,$$

$$\forall f \in F, \forall P \in \mathcal{P}_f \tag{VII.17}$$

$$\sum_{f\in F} \mu_f d_f \geq \frac{p(\boldsymbol{d})}{\lambda^*(\boldsymbol{d})}, \forall \boldsymbol{d} \in \mathcal{D}$$

$$\text{where } \boldsymbol{d} = (d_f, f \in F) \tag{VII.18}$$

Now we present an approximation algorithm for $\mathbf{P_U}$ in Table VII.3. This algorithm (UM^3R) has the same initialization as the algorithm for problem $\mathbf{P_T}$ (FM^3R). Then we march into the iteration, in which we find $\boldsymbol{d}^{\min}$, the demand whose price $\mu^{\min}$ is the minimum among others (Lines 4 to 12). If $\mu^{\min} \geq 1$, then the algorithm stops (Lines 13 and 14), since Inequality (VII.17) and (VII.18) would be satisfied for all demand. Otherwise, we will increase the price of $\boldsymbol{d}^{\min}$ by routing more traffic through its node pairs. This procedure (Lines 16 to 23) is the same as what has been described in Lines 4 to 11 of FM^3R algorithm. Following the same proving sequence for FM^3R, we are able to prove the similar properties with UM^3R.

Theorem 2: If $\beta = ((|E_c| + |V|)/(1-\epsilon))^{-1/\epsilon}$, then the final flow generated by UM^3R is at least $(1-3\epsilon)$ times the optimal value of $\mathbf{P_U}$. The running time is $O(\frac{1}{\epsilon^2}[\log(|E_c| + |V|)(2|\mathcal{D}||F|\log|F| + |E_c| + |V|) + \log U)]) \cdot T_{mp}$, where U is the length of the longest path in G, T_{mp} is the running time to find the shortest path.

UM^3R: Mesh Network Routing Under Uncertain Demand

$\forall e \in E$, $\mu_e \leftarrow \beta$, $\mu_v \leftarrow \beta/\kappa(v)$
$x_f(P) \leftarrow 0$, $\forall P \in \mathcal{P}_f$, $\forall f \in F$
loop
  for $\forall f \in F$ **do**
    $\bar{P} \leftarrow$ lowest priced path in $\mathcal{P}_f$ using μ_e, μ_v
    $\mu_f \leftarrow \sum_{e \in E} A_{e(c)\bar{P}} \mu_e + B_{v\bar{P}} \mu_v$
  end for
  for $\forall \boldsymbol{d} \in \mathcal{D}$ **do**
    $\mu_{\boldsymbol{d}} \leftarrow \sum_{f \in F} \mu_f d_f \frac{\lambda^*(\boldsymbol{d})}{p(\boldsymbol{d})}$
  end for
  $\mu^{\min} \leftarrow \min_{\boldsymbol{d} \in \mathcal{D}} \mu_{\boldsymbol{d}}$
  $\boldsymbol{d}^{\min} \leftarrow \arg\min_{\boldsymbol{d} \in \mathcal{D}} \mu^{\min}$
  if $\mu^{\min} \geq 1$
    return
  for $\forall f \in F$ **do**
    $d'_f \leftarrow d^{\min}_f$
    while $d'_f > 0$ **do**
      $P \leftarrow$ lowest priced path in $\mathcal{P}_f$ using μ_e, μ_v
      $\delta \leftarrow \min\{d'_f, \min_{e(c) \in P} \frac{1}{A_{e(c)P}}, \min_{v \in V} \frac{\kappa(v)}{B_{vP}}\}$
      $d'_f \leftarrow d'_f - \delta$
      $x_f(P) \leftarrow x_f(P) + \delta$
      $\forall e(c) \in E_c$ s.t. $A_{e(c)P} \neq 0$, $\mu_e \leftarrow \mu_e(1 + \epsilon\delta A_{e(c)P})$
      $\forall v \in V$ s.t. $B_{vP} \neq 0$, $\mu_v \leftarrow \mu_v(1 + \epsilon\delta B_{vP}/\kappa(v))$
    end while
  end for
end loop

Table VII.3: Routing Algorithm Under Uncertain Demand

Simulation Study

Part of the content in section is similar to the simulation part of the previous chapter. However, in order to make this chapter as a self-contained chapter, we still keep this content for ease of reading.

To realistically simulate the traffic demand at each LAP, we employ the traces collected in the campus wireless LAN network. The network traces used in this work are collected in Spring 2002 at Dartmouth College and provided by CRAWDAD [100]. By analyzing the *snmp* log trace at each access point, we are able to derive its 1108-hour incoming and outgoing traffic volume since 12:00AM, March 25, 2002 EST. We select the access points from the Dartmouth campus wireless LAN and assign their traffic traces to the LAPs in our simulation.

We evaluate and compare different traffic prediction and routing strategies for this simulated network. In particular, we consider the following strategies.

- *Perfect Routing (PR).* In this strategy, the traffic demand is known a priori. It runs the FM^3R algorithm (presented in Tab. VII.2) based on this demand. This solution runs every hour based on the up-to-date traffic demand from the trace and returns the optimal set of routes. This ideal strategy is designed to return the benchmark result, which the rest of the practical strategies compare to.

- *Single-Value Prediction Routing (SVPR).* This strategy does not know the traffic demand a priori. Instead, it only predicts the traffic demand based on its historical data. In particular, it employs the mean value prediction model and runs the FM^3R algorithm based on this predicted demand. This solution also runs every hour to provide the set of routes for the next hour.

- *Prediction with Distribution Routing (PDR).* Similar to SVPR, this strategy also relies on traffic prediction. It predicts not only the mean-value of the traffic demand in the next hour, but also its distribution. It runs the UM^3R algorithm (presented in Tab. VII.3) with the predicted traffic demand distribution as its input. Since UM^3R only accepts discrete probability distribution, we need to discretize the demand distribution by sampling the following values, the mean value μ, and values $\mu - \sigma$, $\mu + \sigma$, $\mu - 2\sigma$, and $\mu - 2\sigma$. Since about 95% of all traffic demand values fall within the range $[\mu - 2\sigma, \mu + 2\sigma]$, we ignore the values which has a probability smaller than 5%.

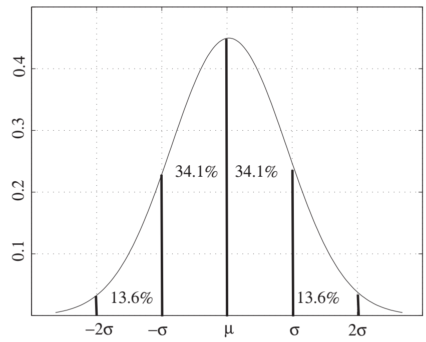


Figure VII.1: Discretization of Traffic Distribution

As illustrated in Fig. VII.1,

- *Shortest-Path Routing (SPR).* This strategy is agnostic of traffic demand, and returns fixed routing solution purely based on the shortest distance (number of hops) from each mesh node to the gateway. The purpose to evaluate this strategy is to quantitatively contrast the advantage of our traffic-predictive routing strategies.

Note that the flows derived from the above routing strategies will be adjusted by the channel assignment, post processing and flow scaling algorithms in [18]. We denote the final rate of flow f along path P as $x_f^A = \sum_{P \in \mathcal{P}_f} x_f^A(P)$. This is the maximum flow throughput under the fairness constraint weighted by the traffic demand, which maximizes the scaling factor λ. However, for performance study, λ is not a suitable performance metric. First, we are more interested in the network performance (*i.e.*, congestion) incurred by the given traffic demand, instead of the achievable throughput. Second, the absolute value of λ could be misleading, especially when the actual demand is not the same as the predicted demand which is being used for routing.

Now we proceed to define the performance metric we use in the simulation study. First, we scale the achievable flow rate x_f^A derived from the routing and channel assignment process by its actual traffic demand d_f:

$$x'_f(P) = x_f^A(P) \cdot \frac{d_f}{x_f^A} \tag{VII.19}$$

$x'_f(P)$ is the actual traffic load that is imposed on path P under our routing and channel assignment scheme. Thus the traffic being routed within the interference set $I_e(c)$ over channel c is given by $\sum_{f \in F} \sum_{P \in \mathcal{P}_f} x'_f(P) A_{e(c)P}$. We define the *congestion of an interference set* $I_e(c)$ using its utilization and denote it as $\theta_e^{ch}(c) = \frac{\sum_{f \in F} \sum_{P \in \mathcal{P}_f} x'_f(P) A_{e(c)P}}{\gamma(\Delta)}$. Then $\theta^{ch} = \max_{e(c) \in E_c} \theta_e^{ch}(c)$ is the maximum congestion among all the interference sets. We further consider the congestion at a single mesh node incurred by the traffic from all channels. The *congestion of a node* v is defined as $\theta_v^{rd} = \frac{\sum_{f \in F} \sum_{P \in \mathcal{P}_f} x'_f(P) B_{vP}}{\kappa(v)}$. And $\theta^{rd} = \max_{v \in V} \theta_v^{rd}$. Finally, the *network congestion* θ is defined as $\theta = \max\{\theta^{rd}, \theta^{ch}\}$.

Simulation on Self-developed Simulator

This section describes the simulation results from the self-developed simulator. In the simulated wireless mesh network, 60 mesh nodes are randomly deployed over a $1000 \times 2000m^2$ region. 20 nodes at the edge of this network are selected as the local access points (LAP) that forward traffic for clients. 4 nodes in the center of the deploy region are selected as the gateway access points. Each mesh node has a transmission range of $250m$ and an interference range of $500m$, which means $\Delta = 2$. The channel capacity $\phi_c(e)$ is the same for all links e and channels c, which is set as 54 Mbps. In the basic setting, each mesh nodes are equipped with 3 radio interfaces. And there are 3 orthogonal channels in the network. Aside from this basic setting, we have also evaluated the performance of our algorithms with different configurations of radio and channel numbers, which we will show in the later part of this section.

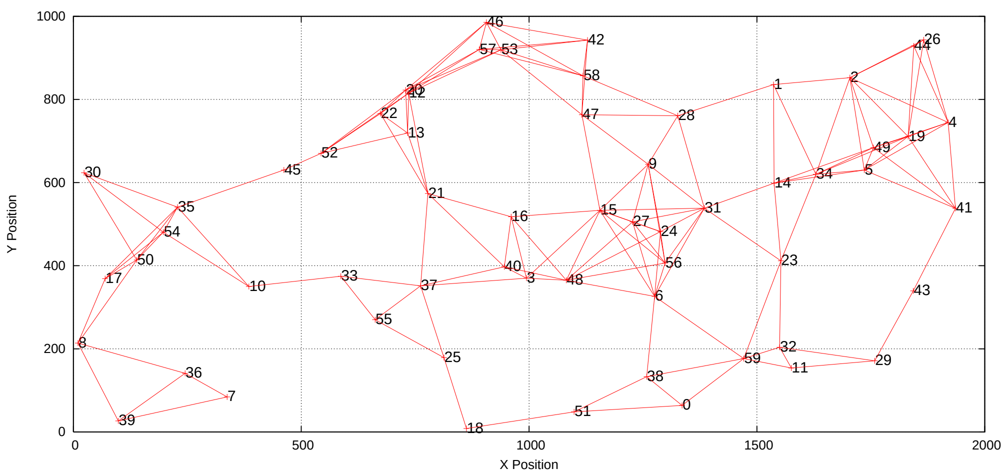


Figure VII.2: Mesh Network Topology.

AP	31AP3	34AP5	55AP4	57AP2	62AP3	62AP4	82AP4	94AP1	94AP3	94AP8
Node ID	22	18	57	5	55	20	53	3	56	27
AP	27AP3	3AP3	21AP2	23AP4	33AP2	62AP2	82AP3	84AP1	90AP2	97AP2
Node ID	9	23	25	33	19	35	58	42	6	48

Table VII.4: Node Assignment in Self-developed Simulator

We experiment with the above routing strategies along the time range $[108, 1108]$, a 1000-hour period excerpted from the trace[1]. Note that all the simulation results presented in this section are using 108 as the zero point.

We start by presenting the congestion achieved by all strategies (*PR*, *SVPR*, *PDR*, and *SPR*) during the entire 1000-hour simulation period. The overall simulation result shows that *PR* constantly achieves the minimum worst-case congestion among others, due to its unrealistic capability to know the actual traffic demand. We note that the burstiness of θ applies to all strategies including *PR*. Such observation comes from the burstiness of the traffic load in the *snmp* log trace, which is caused by the insufficient level of traffic multiplexing at wireless local access points.

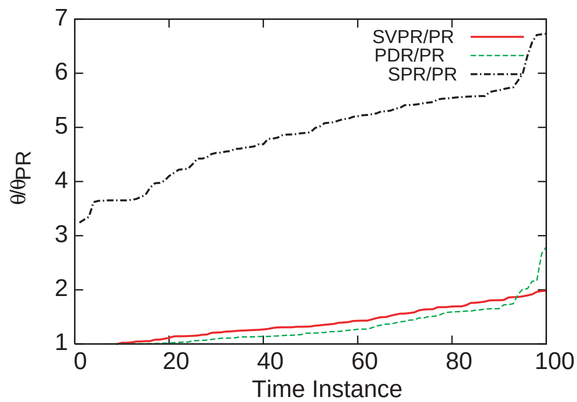


Figure VII.3: Sorted View of Congestion Ratio

[1]Note that the beginning part of the trace $[0, 107]$ is used as training data, thus is not included in the simulation result.

To filter out the noise caused by traffic burstiness, we normalize θ achieved by other strategies by the same value of *PR*. Since *PR* always achieves the minimum θ among others, this ratio will end up at least 1. Also we take a close-up look during the hour range $[190, 290]$. We sort out the normalized congestion ratio for the three strategies in Fig. VII.3. Here, the *SVPR* and *PDR* strategies achieve less than 2 times of the optimal congestion in most cases, while the *SPR* strategy can only achieve $4-7$ times of the optimal performance. It is clear that our *SVPR* and *PDR* strategies which integrate the traffic prediction with the optimal routing outperform the *SPR* strategy which is agnostic about the traffic demand. Further, *PDR* achieves lower congestion than *SVPR* in most of the time due to more comprehensive representation of the traffic demand estimation. However, in a few cases (less than 10% of the time), the worst-case congestion of *PDR* is higher than *SVPR*. This problem can be mostly attributed to the inaccuracy of traffic prediction.

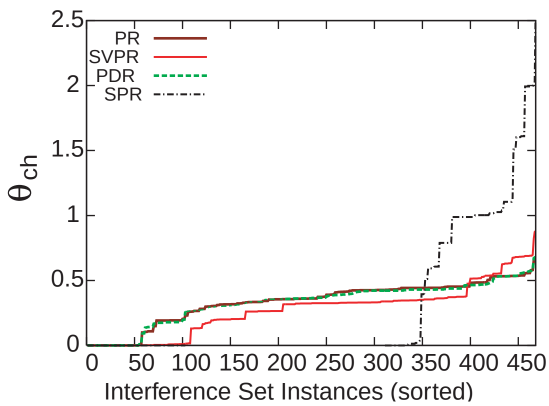
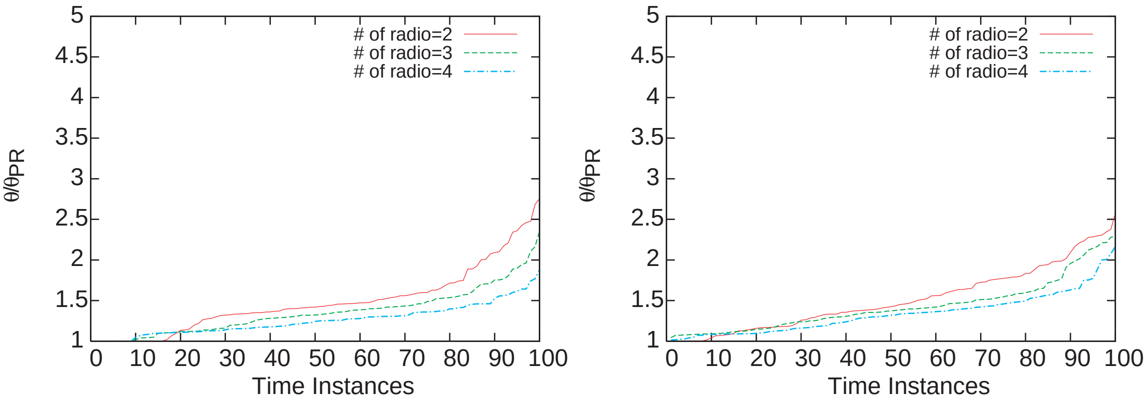


Figure VII.4: Adjusted Interference Set Sorted By Congestion

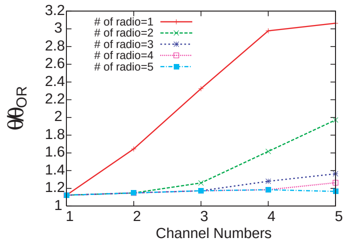
Next, we take a closer look at each strategy's ability to balance the traffic within the mesh network. In Fig. VII.4, we unfold a single time instance at hour 271 and exhibit the congestion $\theta^{ch}_{e(c)}$ at each interference set $I_e(c)$ resulted from each strategy. In order to achieve the lowest worst-case congestion, a good strategy should maximally even out the traffic routed through all interference sets. Obviously, *PR* achieves such a balance, which resulted in the best θ value 0.65. *SPR* has the highest θ value as more than 2. The results for *SVPR* and *PDR* are 0.8 and 0.7 respectively. We also observe that the distribution of $\theta^{ch}_{e(c)}$ under the *PDR* strategy closely matches the *PR* strategy.



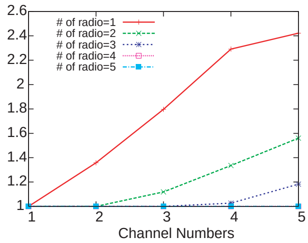
(a) SVPR (b) PDR

Figure VII.5: Impact of Number of Radio Interfaces

In what follows, we alter our simulation configurations to examine the abilities of different strategies at adapting various network settings, such as radio interface numbers and channel numbers. Here, we focus on the traffic prediction strategies, namely, *SVPR* and *PDR*. Also we plot their performances by the congestion ratio θ/θ_{PR} normalized by the *PR* routing results. We first vary the number of radio interfaces from 2 to 4 and study the congestion θ during the time interval $[190, 290]$. Fig. VII.5 plots the sorted normalized congestion $\frac{\theta}{\theta_{PR}}$ of the two strategies. Comparing these two figures, we could see that the *PDR* strategy performs slightly better than the $SVPR$ strategy. The improvement of both strategies over the PR strategy increases ($i.e.$, normalized congestion decreases) with the radio number.



(a) SVPR



(b) PDR

Figure VII.6: Impact of Channel/Radio

Finally, Fig. VII.6 plots the normalized congestion under different radio and channel numbers at a single time instance 271 for these two strategies. The results show that the improvement of both strategies over the PR strategy decreases with the channel number. This is because when the network has more channels, the algorithms are likely to find more paths and the prediction error is more likely to be magnified.

Simulation on NS2

The previous section shows the simulation results from the self-developed simulation tool, and in this section we present the results from further implement of our algorithms on NS2. Due to some limitations in NS2, we have to make changes to make it better serve for the simulation purpose.

We simulate our algorithms in a wireless mesh network with 64 mesh nodes which are randomly distributed in a $1000 \times 1000m^2$ area. We define that each node has a transmission range and an interference range of both $155m$. All other settings use the default values set in NS2.

We pick two nodes as gateways for the simulation, and ten nodes as LAPs. We still use the same set of traffic generated from Dartmouth College. However, due to the computation complexity, we only use a portion of trace (the hour range $[190, 290]$) for the simulation in NS2.

AP	31AP3	34AP5	55AP4	57AP2	62AP3	62AP4	82AP4	94AP1	94AP3	94AP8
Node ID	28	54	34	43	26	27	49	58	23	13

Table VII.5: Overview of Traffic Demand Assignment in NS2

The simulation on NS2 makes it possible to implement our routing algorithms as well as other existing routing algorithms. By using the same configuration on NS2, we can compare our algorithms performance with those algorithms. In this simulation, we compare some popular routing algorithms (*AODV* and *ETX* [33]). Please note that in our simulation, we have multiple gateways and LAPs can have multiple flows to route the traffic to any gateway. *AODV* and *ETX*, on the other hand, does not support this mode directly and these routing algorithms can only support a single flow solution, so they will choose an optimal routing path to one of gateways based on their own metric.

In the previous simulation that is based on the self-developed simulator, we use θ as the metric to compare how congested the network can be after applying different routing algorithms. In the NS2 simulation, however, it is not possible to measure this metric at each interference set directly. Instead of using θ, we use the original definition of the scaling factor λ in the formulation. It is obvious that, the larger the value of λ is, the better the performance of the algorithm is. We find that there is zero demand traffic in some APs during certain time in the original trace, which could lead λ to 0, and we believe it should be ignored from our algorithm comparison.

We simulate both lossy and lossless network environments. The first set of our simulation assumes that all links are stable, and there is no packet loss due to noises from external environment. Since *ETX* uses a metric that is based on the link loss, the performance comparison in this set of simulations does not include this routing algorithm.

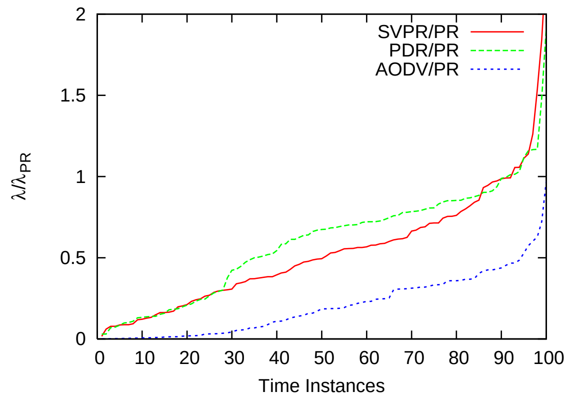


Figure VII.7: Sorted View of Scaling Factor Ratios

Fig. VII.7 is an overview of scaling factor λ using different routing algorithms during the hour range $[190, 290]$ after we normalize the values of routing algorithms by the corresponding ones of *PR* and sort. We can see that *PR*, *SVPR* and *PDR* have much higher values of the scaling factor. Over 20% of the time in the simulation, *SVPR* and *PDR*, especially *PDR*, have performance that is at least 70% of the optimal solution. Please note that there is $5-10\%$ of the time, our algorithms are better than the optimal, which we believe is the gap generated by the difference between algorithm modeling and real network environment.

In Fig. VII.7, we find that *SVPR* and *PDR* have higher throughput than *AODV*. However, the performance between these two algorithms is not clear. In Fig. VII.8, we compare the simulation results by normalize *PDR* using *SVPR*. It is clear to

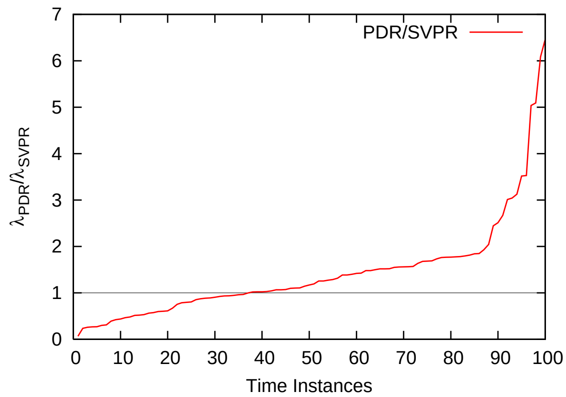


Figure VII.8: Comparison between SVPR and PDR

see that more than 60% of the time, PDR outperforms $SVPR$ and over 10% of the time the performance of PDR is at least twice as good as $SVPR$.

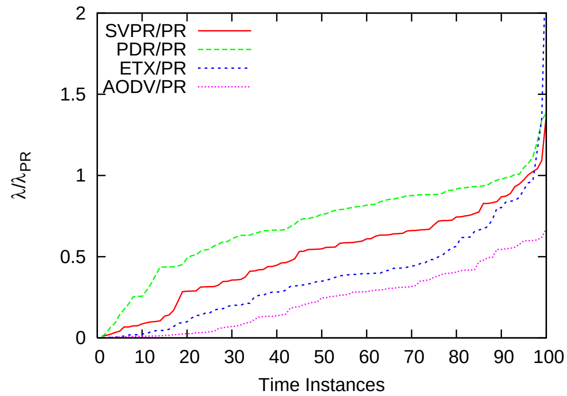


Figure VII.9: Sorted View of Scaling Factor Ratios under Normal Distribution with 20% Average Link Loss

In the second set of the simulation, we simulate algorithms under several lossy environments. The link loss has certain correlation with the link distance, and longer links are more likely to lose packets than shorter ones. We create three different scenarios in order to compare simulation results under different environments. In our first two scenarios, we assume the link distances in our simulation follow a normal distribution and assign loss ratios in proportion to link distances.

We assume the average loss ratio is 40% in the first scenario and 20% in the second one. At the same time, we understand that link distance is not the determined factor for the link loss and there are also other factors that can lead to the packet loss. So in our third scenario, we assign loss ratio randomly to the links with an average loss ratio of 20%.

Fig. VII.9 shows the simulation results from the second scenario where link loss follows link distance and has an average of 20%. We find that both *SVPR* and *PDR* generally have higher performance than *ETX* and *AODV*. We also run our algorithms under two other network environments. When comparing the first two scenarios (normal distribution with 40% and 20% average link loss respectively), we find *AODV* algorithm, which does not consider loss ratio, has the similar performance whenever the loss ratio is high or low when comparing to optimal routing algorithm. *SVPR*, *PDR* and *ETX* have better performance when the loss ratio is high, which confirms that those routing algorithms are adaptive to lossy network environment especially when the link loss becomes critical. When comparing the second scenario to the third scenario where it has the same link loss but assigns randomly, we find that routing algorithms performs better in a simulated environment where link loss is linked to link distance, which also supports our assumption that it is meaningful to assign link loss based on link length.

The packet delay from a sender to a receiver is another important metric for routing algorithms. We trace the time when packets are sent and received and analyze the delay of each packet. A typical simulation result from different scenarios is shown in Fig. VII.10 where link loss is distributed according to link length with an average loss ratio of 20%. In this figure, we compare the delay of all other routing algorithms to that of *AODV* which we believe should generally have longer delay because less-planned flows in *AODV* may congest the network and it may take

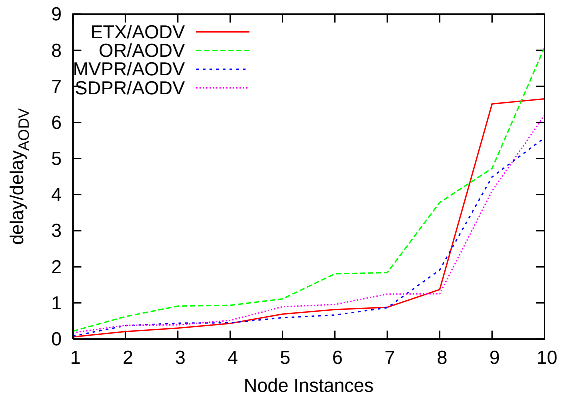


Figure VII.10: Sorted View of Packet Delay at Hour 245

longer time to deliver the packet. The results from Fig. VII.10 show that although *AODV* tries to use the shortest path, it does not necessarily lead to the shortest delay. About 80% of the time all other algorithms except *OR* have shorter delay than *AODV*. The main reason that *OR* has the similar delay as *AODV* is that *OR* tends to find more links to route traffic in order to reach the optimal value, of which some have longer delay. *SVPR*, *SPDR* and *ETX* have similar performance in the simulation and we can not find obvious delay difference among those algorithms.

CHAPTER VIII

CONCLUSION

The WMN is gaining increasing popularity as a high-performance and low-cost solution to last-mile broadband Internet access and disaster recovery. Traffic distribution plays a critical role in determining WMN performance. This dissertation studies a framework that integrates traffic distribution with traffic prediction. It is an optimization-based traffic distribution solution for multi-radio, multi-channel wireless mesh networks, which takes into account the dynamic nature of wireless traffic demand.

The traffic analysis component of the framework establishes traffic models for wireless access points and uses them to predict the future traffic demand for load distribution. We characterize historical traffic using time series models, and predict the future traffic demand based on the established models. We propose two different traffic prediction models. The single value prediction provides the expected value for predicted traffic demand, while prediction with statistic distribution provides possible traffic values with their corresponding probabilities. Traffic distribution algorithms that utilize the probabilistic traffic distribution information is more resilient to traffic dynamics.

In the traffic distribution component, we develop optimization-based algorithms to balance the traffic load. The fixed-demand routing algorithm takes the single-value prediction result and computes the optimal routing based on the determined traffic information. The uncertain demand routing takes the prediction with distribution traffic model as the input and compute the routing that optimizes the expected network performance. We study our routing algorithms under three different network models. The first network model only has a single channel. The routing algorithms developed under this model serve as the baseline algorithms for the other two network models. The second network model incorporates the existence of multiple radios and multiple channels. The baseline routing algorithms are extended to joint solution of channel assignment and routing solutions. The third network model considers the wireless random losses in traffic distribution. Link loss aware routing algorithms can accurately reflect physical network environment and design routing paths by adapting this feature.

The contributions of this dissertation are as follows. First, the integration of traffic estimation and distribution optimization effectively improves the performance of WMNs under dynamic and uncertain traffic. Unlike most existing routing algorithms that ignore the fact of dynamic traffic and lead to poor results when implemented in real networks, our algorithms are adaptive to dynamic traffic, and can improve wireless network performance under uncertain traffic. The full-fledged simulation study based on real wireless network traffic traces provide convincing validation of the practicability of our solution. Second, we extend the classical linear network optimization algorithm, which only accepts the fixed-value demand as input, into a stochastic optimization solution capable of serving uncertain demands that are modeled by their statistical distributions. The results show that our algorithms can achieve better performance under dynamic network environment than classical routing algorithms.

APPENDIX A

APPENDIX I

Proof for Theorem 1

The proof to **Theorem 1** is precluded by a sequence of lemmas. We first make the following denotations. We use OPT to represent the optimal solution of both $\mathbf{P_T}$ and $\mathbf{D_T}$, and OPT' to represent the solution derived from **FMR** algorithm.

Lemma 1 : If $OPT \geq 1$, scaling the final flow by $\log_{1+\epsilon} 1/\beta$ yields a feasible primal solution of value $OPT' = \frac{t-1}{\log_{1+\epsilon} 1/\beta}$, t being the number of phases the algorithm takes to stop.

Proof. We first make the following denotations. Regarding a set of price assignments μ_e for S_e ($e \in E$), the objective function of $\mathbf{D}$ is $L^{\mu_e} \triangleq \sum_{e\in E} c{\cdot}\mu_e$. Let $P^{\mu_e}(f)$ be the minimum path of the flow $f \in F$ using μ_e. $\mu(P^{\mu_e}(f)) \triangleq \sum_{e\in E} A_{eP^{\mu_e}(f)}\mu_e$ is the aggregated price of $P^{\mu_e}(f)$. Each phase contains $|F|$ iterations, where traffic for each flow in F is routed according to its demand. In each iteration, the price of an interference set is updated. We use $\mu_e^{(i)(j)}$ to denote the price of S_e for $e \in E$ after the jth iteration of the ith phase. Regarding $\mu_e^{(i)(j)}$, we simplify the notation $L^{\mu_e^{(i)(j)}}$ into $L^{(i)(j)}$, $P^{\mu_e^{(i)(j)}}$ into $P^{(i)(j)}$, and $\mu(P^{\mu_e^{(i)(j)}})$ into $\mu(P^{(i)(j)})$. Based on the price update function (Line 11 in Tab. V.2), we have

$$
\begin{aligned}
& L^{(i)(j)} \\
= \; & \sum_{e\in E} \mu_e^{(i)(j-1)} + \epsilon \sum_{e\in P^{(i)(j-1)}} A_{eP^{(i)(j-1)}} \mu_e^{(i)(j-1)} d(f_j) \\
= \; & L^{(i)(j-1)} + d(f_j)\mu(P^{(i)(j-1)})
\end{aligned}
$$

The price assignment at the start of the $(i+1)$th phase are the same as that at the end of the ith phase, i.e., $\mu_e^{(i+1)(0)} = \mu_e^{(i)(|F|)}$. The price of any interference set S_e is initialized as $\mu_e^{(1)(0)} = \mu_e^{(0)(|F|)} = \beta/c$. Hence,

$$L^{(i)(|F|)} \leq L^{(i)(0)} + \epsilon \sum_{j=1}^{|F|} d(f_j)\mu(P^{(i)(|F|)})$$

since the edge lengths are monotonically increasing.

Let us define $\mu^{(i)(|F|)} = \sum_{j=1}^{|F|} d(f_j)\mu(P^{(i)(|F|)})$. Then the objective of $\mathbf{D_T}$ is to minimize $L^{(i)(|F|)}$, subject to the constraint that $\mu^{(i)(|F|)} \geq 1$. This constraint can be easily satisfied if we scale the length of all inference sets by $1/\mu^{(i)(|F|)}$. So $\mathbf{D_T}$ is equivalent to finding a set of inference set lengths, such that $\frac{L^{(i)(|F|)}}{\mu^{(i)(|F|)}}$ is minimized. Thus the optimal value of $\mathbf{D_T}$ is $OPT \triangleq \min_{\mu^{(i)(|F|)}} \frac{L^{(i)(|F|)}}{\mu^{(i)(|F|)}}$.

Since $\frac{L^{(i)(|F|)}}{\mu^{(i)(|F|)}} \geq OPT$, we have

$$L^{(i)(|F|)} \leq \frac{\beta|E|}{1-\epsilon} e^{\frac{\epsilon(i-1)}{OPT(1-\epsilon)}}$$

Since $L^{(0)(|F|)} = \beta|E|$, we have

$$\begin{aligned} L^{(i)(|F|)} &\leq \frac{\beta|E|}{(1-\epsilon/OPT)^i} \\ &= \frac{\beta|E|}{(1-\epsilon/OPT)}(1+\frac{\epsilon}{OPT-\epsilon})^{i-1} \\ &\leq \frac{\beta|E|}{(1-\epsilon/OPT)} e^{\frac{\epsilon(i-1)}{OPT-\epsilon}} \\ &\leq \frac{\beta|E|}{1-\epsilon} e^{\frac{\epsilon(i-1)}{OPT(1-\epsilon)}} \end{aligned}$$

where the last inequality assumes that $OPT \geq 1$. The algorithm stops at the first phase t for which $L^{(t)(|F|)} \geq 1$. Therefore,

$$1 \leq L^{(t)(|F|)} \leq \frac{\beta|E|}{1-\epsilon} e^{\frac{\epsilon(t-1)}{OPT(1-\epsilon)}}$$

which implies

$$\frac{OPT}{t-1} \leq \frac{\epsilon}{(1-\epsilon)\ln\frac{1-\epsilon}{\beta|E|}} \tag{A.1}$$

Now consider an interference set S_e. For every c units of flow routed through S_e, we increase its price by at least a factor $(1+\epsilon)$. Initially, its length is β/c and after $t-1$ phases, since $L^{(t)(|F|)} < 1$, the price of S_e satisfies $\mu_e^{(t-1)(|F|)} < 1/c$. Therefore the total amount of flow through S_e in the first $t-1$ phases is strictly less than $\log_{1+\epsilon}\frac{1/c}{\beta/c} = \log_{1+\epsilon} 1/\beta$ times its capacity. Thus, scaling the flow by $\log_{1+\epsilon} 1/\beta$ will yield a feasible solution. Since in each phase, $d(f)$ units of data are routed for each flow, we have $OPT' = \frac{t-1}{\log_{1+\epsilon} 1/\beta}$.

□

Lemma 2: If $OPT \geq 1$, then the final flow scaled by $\log_{1+\epsilon} 1/\beta$ has a value at least $(1-3\epsilon)$ times OPT, when $\beta = (|E|/(1-\epsilon))^{-1/\epsilon}$.

Proof. By **Lemma 1**, scaling the final flow by $\log_{1+\epsilon} 1/\beta$ yields a feasible solution. Therefore,

$$\frac{OPT}{OPT'} < \log_{1+\epsilon} 1/\beta \tag{A.2}$$

Substituting the bound on $OPT/(t-1)$ from In Equality (B.1), we get

$$\frac{OPT}{OPT'} < \frac{\epsilon \log_{1+\epsilon} 1/\beta}{(1-\epsilon)\ln\frac{1-\epsilon}{\beta|E|}} = \frac{\epsilon}{(1-\epsilon)\ln(1+\epsilon)} \frac{\ln 1/\beta}{\ln\frac{1-\epsilon}{\beta|E|}}$$

When $\beta = (|E|/(1-\epsilon))^{-1/\epsilon}$, the above in Equality becomes

$$\begin{aligned}\frac{OPT}{OPT'} &< \frac{\epsilon}{(1-\epsilon)^2\ln(1+\epsilon)} \leq \frac{\epsilon}{(1-\epsilon)^2(\epsilon-\epsilon^2/2)}\frac{1}{\leq (1-\epsilon)^3} \\ &\leq (1-3\epsilon)\end{aligned}$$

□

Lemma 3: If $OPT \geq 1$ and $\beta = (|E|/(1-\epsilon))^{-1/\epsilon}$, **Algorithm I** terminates after at most $t = 1 + \frac{OPT}{\epsilon}\log_{1+\epsilon}\frac{|E|}{1-\epsilon}$ phases.

Proof. From In Equality (B.2) and weak-duality, we have

$$1 \leq \frac{OPT}{OPT'} < \log_{1+\epsilon} 1/\beta$$

Hence, the number of phases t is strictly less than $1 + OPT\log_{1+\epsilon} 1/\beta$. If $\beta = (|E|/(1-\epsilon))^{-1/\epsilon}$, then $t \leq 1 + \frac{OPT}{\epsilon}\log_{1+\epsilon}\frac{|E|}{1-\epsilon}$ □

These lemmas require that $OPT \geq 1$. The running time of the algorithm also depends on OPT. Thus we need to ensure that OPT is at least one and not too large. Let ζ_i be the maximum traffic value of flow f_i when all other flows have zero traffic. Let $\zeta = \min_i \frac{\zeta_i}{d(f_i)}$. Since at best all single commodity maximum flows can be routed simultaneously, ζ is an upper bound on OPT'. On the other hand, routing $1/|F|$ fraction of each flow of value ζ_i is a feasible solution, which implies that $\zeta/|F|$ is a lower bound on OPT. To ensure that $OPT \geq 1$, we can scale the original demands so that $\zeta/|F|$ is at least one. However, by doing so, OPT might be made as large as $|F|$, which is also undesirable.

To reduce the dependence on the number of phases on OPT, we adopt the following technique. If the algorithm does not stop after $T = \frac{2}{\epsilon}\log_{1+\epsilon}\frac{|E|}{1-\epsilon}$ phases, it means that $OPT > 2$. We then double demands of all commodities, so that OPT is halved and still at least 1. We then continue the algorithm, and double demands again if it does not stop after T phases.

These lemmas require that $OPT \geq 1$. The running time of the algorithm also depends on OPT. Thus we need to ensure that OPT is at least one and not too large. Let ζ_f be the maximum traffic value of flow f when all other flows have zero traffic. Let $\zeta = \min_f \frac{\zeta_f}{d_f}$. Since at best all single commodity maximum flows can be routed simultaneously, ζ is an upper bound on OPT'. On the other hand, routing $1/|F|$ fraction of each flow of value ζ_f is a feasible solution, which implies that $\zeta/|F|$ is a lower bound on OPT. To ensure that $OPT \geq 1$, we can scale the original demands so that $\zeta/|F|$ is at least one. However, by doing so, OPT might be made as large as $|F|$, which is also undesirable.

To reduce the dependence on the number of phases on OPT, we adopt the following technique. If the algorithm does not stop after $T = \frac{2}{\epsilon} \log_{1+\epsilon} \frac{|E|}{1-\epsilon}$ phases, it means that $OPT > 2$. We then double demands of all commodities, so that OPT is halved and still at least 1. We then continue the algorithm, and double demands again if it does not stop after T phases.

Lemma 4: Given ζ_f for each flow f, the running time of **Algorithm I** is $O(\frac{\log |E|}{\epsilon^2}(2|F| \log |F| + |E|)) \cdot T_{mp}$.

Proof. The above demand-doubling procedure is repeated for at most $\log |F|$ times. Thus, the total number of phases is at most $T \log k$. Since each phase contains k iterations, the algorithm runs for at most $kT \log k$ iterations.

Now we observe how many steps are within each iteration. For each step except for the last step in an iteration, the algorithm increases the length of some edge

(the bottleneck edge on t) by $1+\epsilon$. d_e has initial value β/c and value at most $1/c$ before the final step of the algorithm. Otherwise, the stop criterion of the algorithm, $\sum_{e \in E} c \cdot d_e \geq 1$, would have been reached. This means that the length of an edge can be updated in at most $\log_{1+\epsilon} \frac{1}{\beta} = \frac{1}{\epsilon} \log_{1+\epsilon} \frac{|E|}{1-\epsilon}$ steps. Therefore, the algorithm contains at most
$\frac{|E|}{\epsilon} \log_{1+\epsilon} \frac{|E|}{1-\epsilon} \leq \frac{|E|}{\epsilon^2} \log \frac{|E|}{1-\epsilon}$ such "normal" steps, and $kT \log k \leq \frac{2k \log k}{\epsilon^2} \log \frac{|E|}{1-\epsilon}$ "final" steps. Each step contains a minimum overlay spanning tree operation. □

Theorem 1: The total running time of **Algorithm I** is $O(\frac{1}{\epsilon^2}[\log |E|(2|F| \log |F| + |E|) + \log U)]) \cdot T_{mp}$.

Proof. Computing ζ_i corresponds to the maximum flow problem, where f_i is the only commodity. The running time of getting ζ_i is $O(\frac{|E|}{\epsilon^2}(\log U)) \cdot T_{mp}$, where U is the length of the longest unicast route, and T_{mp} denotes the running time to find the minimum path. Such an operation has to be repeated for each flow. Also from the result of **Lemma 4**, we can obtain the total running time as described by the theorem. □

Proof for Theorem 2

The proof for **Theorem 2** follows the same sequence as the proof to **Theorem 1**, with minor modification. We start with **Lemma 1**. Each phase of the algorithm contains $|F|$ iterations, where traffic for each flow in F is routed according to its demand. We reuse the same denotations defined in the original proof to **Lemma 1**. We further introduce $\boldsymbol{d}^{(i)}$ as the demand vector chosen at the ith phase.

Based on the price update function (Line 11 in Tab. V.2), we have

$$
\begin{aligned}
& L^{(i)(j)} \\
= \quad & L^{(i)(j-1)} + d(f_j)\mu(P^{(i)(j-1)})\frac{\lambda^*(\boldsymbol{d}^{(i)})}{p(\boldsymbol{d}^{(i)})}
\end{aligned}
$$

The price assignment at the start of the $(i+1)$th phase are the same as that at the end of the ith phase, i.e., $\mu_e^{(i+1)(0)} = \mu_e^{(i)(|F|)}$. The price of any interference set S_e is initialized as $\mu_e^{(1)(0)} = \mu_e^{(0)(|F|)} = \beta/c$. Hence,

$$
\begin{aligned}
L^{(i)(|F|)} &= L^{(i)(0)} + \epsilon \sum_{j=1}^{|F|} d(f_j)\mu(P^{(i)(j-1)})\frac{\lambda^*(\boldsymbol{d}^{(i)})}{p(\boldsymbol{d}^{(i)})} \\
&\leq L^{(i)(0)} + \epsilon \sum_{j=1}^{|F|} d(f_j)\mu(P^{(i)(|F|)})\frac{\lambda^*(\boldsymbol{d}^{(i)})}{p(\boldsymbol{d}^{(i)})}
\end{aligned}
$$

since the edge lengths are monotonically increasing.

Let us define $\mu^{(i)(|F|)} = \sum_{j=1}^{|F|} d(f_j)\mu(P^{(i)(|F|)})\frac{\lambda^*(\boldsymbol{d}^{(i)})}{p(\boldsymbol{d}^{(i)})}$. Then the objective of **D** is to minimize $L^{(i)(|F|)}$, subject to the constraint that $\mu^{(i)(|F|)} \geq 1$, i.e., $\frac{L^{(i)(|F|)}}{\mu^{(i)(|F|)}} \geq OPT$.

The rest of the proof follows the same as the original proof to **Lemma 1**. The proofs to **Lemma 2**, **3**, **4**, and **Theorem 1** remain the same.

APPENDIX B

APPENDIX II

Proof for Theorem 3

The proof to **Theorem 3** is precluded by a sequence of lemmas. We first make the following denotations. We use OPT to represent the optimal solution of both $\mathbf{P_T}$ and $\mathbf{D_T}$, and OPT' to represent the solution derived from FM^3R algorithm.

Lemma 1 : If $OPT \geq 1$, scaling the final flow by $\log_{1+\epsilon} 1/\beta$ yields a feasible primal solution of value $OPT' = \frac{t-1}{\log_{1+\epsilon} 1/\beta}$, t being the number of phases the algorithm takes to stop.

Proof. We first make the following denotations. Regarding a set of price assignments μ_e for $e(c)$ ($e(c) \in E_c$), μ_v for v ($v \in V$), the objective function of $\mathbf{D_T}$ is L^μ. Let $P^\mu(f)$ be the minimum path of the flow $f \in F$ using μ_e and μ_v. $\mu(P^\mu(f)) \triangleq \sum_{e(c)\in E_c} A_{e(c)P^\mu(f)}\mu_e + \sum_{v\in V} B_{vP^\mu(f)}\mu_v$ is the aggregated price of $P^\mu(f)$. Each phase contains $|F|$ iterations, where traffic for each flow in F is routed according to its demand. In each iteration, the price of an interference set is updated. We use $\mu_e^{(i)(j)}$ to denote the price of $e(c) \in E_c$, $\mu_v^{(i)(j)}$ to denote the price of $v \in V$ after the jth iteration of the ith phase. Regarding $\mu_e^{(i)(j)}$ and $\mu_v^{(i)(j)}$, we simplify the notation $L^{\mu^{(i)(j)}}$ into $L^{(i)(j)}$, $P^{\mu^{(i)(j)}}$ into $P^{(i)(j)}$, and $\mu(P^{\mu^{(i)(j)}})$ into

$\mu(P^{(i)(j)})$. Based on the price update function (Line 11 in Tab. VI.2), we have

$$\begin{aligned}
& L^{(i)(j)} \\
= & \sum_{e(c)\in E_c} \mu_e^{(i)(j-1)} + \epsilon \sum_{e(c)\in P^{(i)(j-1)}} A_{e(c)P^{(i)(j-1)}} \mu_e^{(i)(j-1)} d(f_j) \\
+ & \sum_{v\in V} \mu_v^{(i)(j-1)} + \epsilon \sum_{v\in P^{(i)(j-1)}} B_{vP^{(i)(j-1)}} \mu_v^{(i)(j-1)} d(f_j) \\
= & L^{(i)(j-1)} + d(f_j)\mu(P^{(i)(j-1)})
\end{aligned}$$

The price assignment at the start of the $(i+1)$th phase are the same as that at the end of the ith phase, i.e., $\mu_e^{(i+1)(0)} = \mu_e^{(i)(|F|)}$. The price of any interference set $e(c)$ is initialized as $\mu_e^{(1)(0)} = \mu_e^{(0)(|F|)} = \beta/\gamma(\Delta)$, and the price of any node v is initialized as $\mu_v^{(1)(0)} = \mu_V^{(0)(|F|)} = \beta/\kappa(v)$. Hence,

$$L^{(i)(|F|)} \leq L^{(i)(0)} + \epsilon \sum_{j=1}^{|F|} d(f_j)\mu(P^{(i)(|F|)})$$

since μ_e and μ_v are monotonically increasing.

Let us define $\mu^{(i)(|F|)} = \sum_{j=1}^{|F|} d(f_j)\mu(P^{(i)(|F|)})$. Then the objective of $\mathbf{D_T}$ is to minimize $L^{(i)(|F|)}$, subject to the constraint that $\mu^{(i)(|F|)} \geq 1$. This constraint can be easily satisfied if we scale the prices of all inference sets and nodes by $1/\mu^{(i)(|F|)}$. So $\mathbf{D_T}$ is equivalent to finding a set of inference set lengths, such that $\frac{L^{(i)(|F|)}}{\mu^{(i)(|F|)}}$ is minimized. Thus the optimal value of $\mathbf{D_T}$ is $OPT \triangleq \min_{\mu^{(i)(|F|)}} \frac{L^{(i)(|F|)}}{\mu^{(i)(|F|)}}$.

Since $\frac{L^{(i)(|F|)}}{\mu^{(i)(|F|)}} \geq OPT$, we have

$$L^{(i)(|F|)} \leq \frac{L^{(0)(|F|)}}{1-\epsilon} e^{\frac{\epsilon(i-1)}{OPT(1-\epsilon)}}$$

Since $L^{(0)(|F|)} = \beta(|E_c| + |V|)$, we have

$$\begin{aligned} L^{(i)(|F|)} &\leq \frac{\beta(|E_c| + |V|)}{(1 - \epsilon/OPT)^i} \\ &= \frac{\beta(|E_c| + |V|)}{(1 - \epsilon/OPT)}(1 + \frac{\epsilon}{OPT - \epsilon})^{i-1} \\ &\leq \frac{\beta(|E_c| + |V|)}{(1 - \epsilon/OPT)} e^{\frac{\epsilon(i-1)}{OPT-\epsilon}} \\ &\leq \frac{\beta(|E_c| + |V|)}{1 - \epsilon} e^{\frac{\epsilon(i-1)}{OPT(1-\epsilon)}} \end{aligned}$$

where the last inequality assumes that $OPT \geq 1$. The algorithm stops at the first phase t for which $L^{(t)(|F|)} \geq 1$. Therefore,

$$1 \leq L^{(t)(|F|)} \leq \frac{\beta(|E_c| + |V|)}{1 - \epsilon} e^{\frac{\epsilon(t-1)}{OPT(1-\epsilon)}}$$

which implies

$$\frac{OPT}{t-1} \leq \frac{\epsilon}{(1-\epsilon)\ln \frac{1-\epsilon}{\beta(|E_c|+|V|)}} \tag{B.1}$$

Now consider an interference set $e(c)$. For every $\gamma(\Delta)$ units of flow routed through $e(c)$, we increase its price by at least a factor $(1 + \epsilon)$. Initially, its length is $\beta/\gamma(\Delta)$ and after $t - 1$ phases, since $L^{(t)(|F|)} < 1$, the price of $e(c)$ satisfies $\mu_e^{(t-1)(|F|)} < 1/\gamma(\Delta)$. Therefore the total amount of flow through $e(c)$ in the first $t - 1$ phases is strictly less than $\log_{1+\epsilon} \frac{1/\gamma(\Delta)}{\beta/\gamma(\Delta)} = \log_{1+\epsilon} 1/\beta$ times its capacity. The same procedure applies for any node v. Thus, scaling the flow by $\log_{1+\epsilon} 1/\beta$ will yield a feasible solution. Since in each phase, $d(f)$ units of data are routed for each flow, we have $OPT' = \frac{t-1}{\log_{1+\epsilon} 1/\beta}$.

□

Lemma 2: If $OPT \geq 1$, then the final flow scaled by $\log_{1+\epsilon} 1/\beta$ has a value at least $(1-3\epsilon)$ times OPT, when $\beta = ((|E_c| + |V|)/(1-\epsilon))^{-1/\epsilon}$.

Proof. By **Lemma 1**, scaling the final flow by $\log_{1+\epsilon} 1/\beta$ yields a feasible solution. Therefore,

$$\frac{OPT}{OPT'} < \log_{1+\epsilon} 1/\beta \tag{B.2}$$

Substituting the bound on $OPT/(t-1)$ from In Equality (B.1), we get

$$\frac{OPT}{OPT'} < \frac{\epsilon \log_{1+\epsilon} 1/\beta}{(1-\epsilon)\ln \frac{1-\epsilon}{\beta(|E_c|+|V|)}} = \frac{\epsilon}{(1-\epsilon)\ln(1+\epsilon)} \frac{\ln 1/\beta}{\ln \frac{1-\epsilon}{\beta(|E_c|+|V|)}}$$

When $\beta = ((|E_c| + |V|)/(1-\epsilon))^{-1/\epsilon}$, the above in Equality becomes

$$\begin{aligned} \frac{OPT}{OPT'} &< \frac{\epsilon}{(1-\epsilon)^2 \ln(1+\epsilon)} \leq \frac{\epsilon}{(1-\epsilon)^2(\epsilon - \epsilon^2/2)} \frac{1}{\leq (1-\epsilon)^3} \\ &\leq (1-3\epsilon) \end{aligned}$$

□

Lemma 3: If $OPT \geq 1$ and $\beta = ((|E_c| + |V|)/(1-\epsilon))^{-1/\epsilon}$, **Algorithm I** terminates after at most $t = 1 + \frac{OPT}{\epsilon} \log_{1+\epsilon} \frac{|E_c|+|V|}{1-\epsilon}$ phases.

Proof. From In Equality (B.2) and weak-duality, we have

$$1 \leq \frac{OPT}{OPT'} < \log_{1+\epsilon} 1/\beta$$

Hence, the number of phases t is strictly less than $1 + OPT \log_{1+\epsilon} 1/\beta$. If $\beta = ((|E_c| + |V|)/(1-\epsilon))^{-1/\epsilon}$, then $t \leq 1 + \frac{OPT}{\epsilon} \log_{1+\epsilon} \frac{|E_c|+|V|}{1-\epsilon}$ □

These lemmas require that $OPT \geq 1$. The running time of the algorithm also depends on OPT. Thus we need to ensure that OPT is at least one and not too large. Let ζ_i be the maximum traffic value of flow f_i when all other flows have zero traffic. Let $\zeta = \min_i \frac{\zeta_i}{d(f_i)}$. Since at best all single commodity maximum flows can be routed simultaneously, ζ is an upper bound on OPT'. On the other hand, routing $1/|F|$ fraction of each flow of value ζ_i is a feasible solution, which implies that $\zeta/|F|$ is a lower bound on OPT. To ensure that $OPT \geq 1$, we can scale the original demands so that $\zeta/|F|$ is at least one. However, by doing so, OPT might be made as large as $|F|$, which is also undesirable.

To reduce the dependence on the number of phases on OPT, we adopt the following technique. If the algorithm does not stop after $T = \frac{2}{\epsilon}\log_{1+\epsilon}\frac{(|E_c|+|V|)}{1-\epsilon}$ phases, it means that $OPT > 2$. We then double demands of all commodities, so that OPT is halved and still at least 1. We then continue the algorithm, and double demands again if it does not stop after T phases.

These lemmas require that $OPT \geq 1$. The running time of the algorithm also depends on OPT. Thus we need to ensure that OPT is at least one and not too large. Let ζ_f be the maximum traffic value of flow f when all other flows have zero traffic. Let $\zeta = \min_f \frac{\zeta_f}{d_f}$. Since at best all single commodity maximum flows can be routed simultaneously, ζ is an upper bound on OPT'. On the other hand, routing $1/|F|$ fraction of each flow of value ζ_f is a feasible solution, which implies that $\zeta/|F|$ is a lower bound on OPT. To ensure that $OPT \geq 1$, we can scale the original demands so that $\zeta/|F|$ is at least one. However, by doing so, OPT might be made as large as $|F|$, which is also undesirable.

To reduce the dependence on the number of phases on OPT, we adopt the following technique. If the algorithm does not stop after $T = \frac{2}{\epsilon}\log_{1+\epsilon}\frac{(|E_c|+|V|)}{1-\epsilon}$ phases, it means that $OPT > 2$. We then double demands of all commodities, so that OPT is halved and still at least 1. We then continue the algorithm, and double demands again if it does not stop after T phases.

Lemma 4: Given ζ_f for each flow f, the running time of **Algorithm I** is $O(\frac{\log(|E_c|+|V|)}{\epsilon^2}(2|F|\log|F|+|E_c|+|V|))\cdot T_{mp}$.

Proof. The above demand-doubling procedure is repeated for at most $\log|F|$ times. Thus, the total number of phases is at most $T\log|F|$. Since each phase contains $|F|$ iterations, the algorithm runs for at most $|F|T\log|F|$ iterations.

Now we observe how many steps are within each iteration. For each step except for the last step in an iteration, the algorithm increases the price of some edge inference set or node by $1+\epsilon$. μ_e has initial value $\beta/\gamma(\Delta)$ and value at most $1/\gamma(\Delta)$ before the final step of the algorithm. The same condition applies for nodes $v \in V$. Otherwise, the stop criterion of the algorithm would have been reached. This means that the price of an edge inference set or node can be updated in at most $\log_{1+\epsilon}\frac{1}{\beta} = \frac{1}{\epsilon}\log_{1+\epsilon}\frac{|E_c|+|V|}{1-\epsilon}$ steps. Therefore, the algorithm contains at most $\frac{|E_c|+|V|}{\epsilon}\log_{1+\epsilon}\frac{|E_c|+|V|}{1-\epsilon} \leq \frac{|E_c|+|V|}{\epsilon^2}\log\frac{|E_c|+|V|}{1-\epsilon}$ such "normal" steps, and $|F|T\log|F| \leq \frac{2|F|\log|F|}{\epsilon^2}\log\frac{|E_c|+|V|}{1-\epsilon}$ "final" steps. Each step contains a minimum overlay spanning tree operation. □

Theorem 3: The total running time of **Algorithm I** is $O(\frac{1}{\epsilon^2}[\log(|E_c|+|V|)(2|F|\log|F|+|E_c|+|V|)+\log U)])\cdot T_{mp}$.

Proof. Computing ζ_i corresponds to the maximum flow problem, where f_i is the only commodity. The running time of getting ζ_i is $O(\frac{|E_c|+|V|}{\epsilon^2}(\log U))\cdot T_{mp}$, where U is the length of the longest unicast route, and T_{mp} denotes the running time to find the minimum path. Such an operation has to be repeated for each flow. Also from the result of **Lemma 4**, we can obtain the total running time as described by the theorem. □

Proof for Theorem 4

The proof for **Theorem 4** follows the same sequence as the proof to **Theorem 3**, with minor modification. We start with **Lemma 1**. Each phase of the algorithm contains $|F|$ iterations, where traffic for each flow in F is routed according to its demand. We reuse the same denotations defined in the original proof to **Lemma 1**. We further introduce $\boldsymbol{d}^{(i)}$ as the demand vector chosen at the ith phase.

Based on the price update function (Line 11 in Tab. VI.2), we have

$$\begin{aligned} & L^{(i)(j)} \\ = \; & L^{(i)(j-1)} + d(f_j)\mu(P^{(i)(j-1)})\frac{\lambda^*(\boldsymbol{d}^{(i)})}{p(\boldsymbol{d}^{(i)})} \end{aligned}$$

The price assignment at the start of the $(i+1)$th phase are the same as that at the end of the ith phase, i.e., $\mu_e^{(i+1)(0)} = \mu_e^{(i)(|F|)}$. The price of any interference set S_e is initialized as $\mu_e^{(1)(0)} = \mu_e^{(0)(|F|)} = \beta/\gamma(\Delta)$, $\mu_v^{(1)(0)} = \mu_v^{(0)(|F|)} = \beta/\kappa(v)$. Hence,

$$\begin{aligned} L^{(i)(|F|)} &= L^{(i)(0)} + \epsilon\sum_{j=1}^{|F|} d(f_j)\mu(P^{(i)(j-1)})\frac{\lambda^*(\boldsymbol{d}^{(i)})}{p(\boldsymbol{d}^{(i)})} \\ &\leq L^{(i)(0)} + \epsilon\sum_{j=1}^{|F|} d(f_j)\mu(P^{(i)(|F|)})\frac{\lambda^*(\boldsymbol{d}^{(i)})}{p(\boldsymbol{d}^{(i)})} \end{aligned}$$

since the edge lengths are monotonically increasing.

Let us define $\mu^{(i)(|F|)} = \sum_{j=1}^{|F|} d(f_j)\mu(P^{(i)(|F|)})\frac{\lambda^*(\boldsymbol{d}^{(i)})}{p(\boldsymbol{d}^{(i)})}$. Then the objective of $\mathbf{D}$ is to minimize $L^{(i)(|F|)}$, subject to the constraint that $\mu^{(i)(|F|)} \geq 1$, i.e., $\frac{L^{(i)(|F|)}}{\mu^{(i)(|F|)}} \geq OPT$.

The rest of the proof follows the same as the original proof to **Lemma 1**. The proofs to **Lemma 2**, **3** remain the same. In the proof of **Lemma 4**, the total number of phases is changed from at most $T\log|F|$ to $T|\mathcal{D}|\log|F|$. The proof of *Theorem 4* follows these results.

BIBLIOGRAPHY

[1] "MIT Roofnet." http://www.pdos.lcs.mit.edu/roofnet/.

[2] "Ucsb meshnet." http://moment.cs.ucsb.edu/meshnet/.

[3] "Microsoft mesh network." http://research.microsoft.com/en-us/projects/mesh.

[4] "Cisco mesh." http://www.cisco.com/en/US/netsol/ns621/networking_solutions_package.html.

[5] "Tropos networks." http://www.tropos.com/.

[6] "Strix systems." http://www.strixsystems.com/.

[7] "BelAir Networks." http://www.belairnetworks.com/.

[8] I. Akyildiz and X. Wang, "A survey on wireless mesh networks," *Communications Magazine, IEEE*, vol. 43, pp. S23–S30, Sept. 2005.

[9] "IEEE 802.11." http://www.ieee802.org/11/.

[10] E. W. Dijkstra, "A note on two problems in connexion with graphs," *Numerische Mathematik*, vol. 1, pp. 269–271, 1959.

[11] Z. Wang and J. Crowcroft, "Analysis of Shortest-path Routing Algorithms in a Dynamic Network Environment," *Journal of ACM SIGCOMM Computer Communication Review*, vol. 22, no. 2, 1992.

[12] S. Pallottino, "Shortest-path methods: Complexity, interrelations and new propositions," *Networks*, vol. 14, pp. 257–267, 1984.

[13] P. Humblet, "A distributed algorithm for minimum weight directed spanning trees," *Communications, IEEE Transactions on*, vol. 31, pp. 756–762, Jun 1983.

[14] P. Gupta and P. Kumar, "The capacity of wireless networks," *IEEE Trans. on Information Theory*, pp. 388–404, 2000.

[15] R. Draves, J. Padhye, and B. Zill, "Routing in multi-radio, multi-hop wireless mesh networks," in *Proc. of ACM Mobicom*, pp. 114–128, 2004.

[16] S. Biswas and R. Morris, "Exor: opportunistic multi-hop routing for wireless networks," in *Proc. of ACM SIGCOMM*, pp. 133–144, 2005.

[17] A. Raniwala and T. Chiueh, "Architecture and algorithms for an ieee 802.11-based multi-channel wireless mesh network," in *Proc. of IEEE INFOCOM*, 2005.

[18] M. Alicherry, R. Bhatia, and L. Li, "Joint channel assignment and routing for throughput optimization in multi-radio wireless mesh networks," in *Proc. of ACM MobiCom*, 2005.

[19] J. Tang, G. Xue, and W. Zhang, "Maximum throughput and fair bandwidth allocation in multi-channel wireless mesh networks," in *Proc. of IEEE INFOCOM*, 2006.

[20] X. G. Meng, S. H. Y. Wong, Y. Yuan, and S. Lu, "Characterizing flows in large wireless data networks," in *MobiCom '04: Proceedings of the 10th annual international conference on Mobile computing and networking*, (New York, NY, USA), pp. 174–186, ACM, 2004.

[21] C. Lopes, A. de Sousa, and L. Gouveia, "Combined link dimensioning and weight assignment of minimum weight routing networks," pp. 181–187, April 2005.

[22] M. Ericsson, M. G. C. Resende, and P. M. Pardalos, "A genetic algorithm for the weight setting problem in ospf routing," *Journal of Combinatorial Optimization*, vol. 6, pp. 299–333, 2001.

[23] R.E.Bellman, "On a routing problem," *Quarterly of Applied Mathematics*.

[24] J. F. D. R. Ford, L. R., "Flows in networks,"

[25] C. Cheng, R. Riley, S. P. R. Kumar, and J. J. Garcia-Luna-Aceves, "A loop-free extended bellman-ford routing protocol without bouncing effect," *SIGCOMM Comput. Commun. Rev.*, vol. 19, pp. 224–236, August 1989.

[26] B. L. Golden, "A minimum-cost multicommodity network flow problem concerning imports and exports," *Networks*, vol. 5, no. 4, pp. 331–356, 1975.

[27] J. A. Tomlin, "Minimum-cost multicommodity network flows," *Operations Research*, vol. 14, no. 1, pp. 45–51, 1966.

[28] M. D. Grigoriadis and W. W. White, "A partitioning algorithm for the multicommodity network flow problem," *Mathematical Programming*, vol. 3, no. 1, pp. 157–177, 1972.

[29] L. K. L. Fratta, M. Gerla, "The flow deviation method: An approach to store-and-forward communication network design," *Networks*, vol. 3, no. 2, pp. 97–133, 1973.

[30] E. Rosen, A. Viswanathan, and R. Callon, "Multiprotocol Label Switching Architecture." RFC 3031 (Proposed Standard), Jan. 2001.

[31] R. Boorstyn and H. Frank, "Large-scale network topological optimization," *Communications, IEEE Transactions on*, vol. 25, pp. 29–47, Jan 1977.

[32] F. Shahrokhi and D. W. Matula, "The maximum concurrent flow problem," *J. ACM*, vol. 37, no. 2, pp. 318–334, 1990.

[33] D. S. J. De Couto, D. Aguayo, J. Bicket, and R. Morris, "A high-throughput path metric for multi-hop wireless routing," in *Proceedings of the 9th ACM International Conference on Mobile Computing and Networking (MobiCom '03)*, (San Diego, California), September 2003.

[34] R. Draves, J. Padhye, and B. Zill, "Comparison of routing metrics for static multi-hop wireless networks," *SIGCOMM Comput. Commun. Rev.*, vol. 34, no. 4, pp. 133–144, 2004.

[35] H. Wu, F. Yang, K. Tan, J. Chen, Q. Zhang, and Z. Zhang, "Distributed channel assignment and routing in multiradio multichannel multihop wireless networks," *IEEE Journal on Selected Areas in Communications*, vol. 24, pp. 1972–1983, November 2006.

[36] X. Lin and S. Rasool, "A distributed joint channel-assignment, scheduling and routing algorithm for multi-channel ad hoc wireless networks," in *INFOCOM*, 2007.

[37] A. Raniwala, K. Gopalan, and T. Chiueh, "Centralized channel assignment and routing algorithms for multi-channel wireless mesh networks," *Mobile Computing and Communications Review*, vol. 8, no. 2, pp. 50–65, 2004.

[38] M. Kodialam and T. Nandagopal, "Characterizing the capacity region in multi-radio multi-channel wireless mesh networks," in *Proc. of IEEE WiMesh*, 2005.

[39] A. Subramanian, M. Buddhikot, and S. Miller, "Interference aware routing in multi-radio wireless mesh networks," *Wireless Mesh Networks, 2006. WiMesh 2006. 2nd IEEE Workshop on*, pp. 55–63, 2006.

[40] S. Zhao, Z. Wu, A. Acharya, and D. Raychaudhuri, "Parma: A phy/mac aware routing metric for ad-hoc wireless networks with multi-rate radios," *WOWMOM*, vol. 1, pp. 286–292, 2005.

[41] G. Jakllari, S. Eidenbenz, N. Hengartner, S. Krishnamurthy, and M. Faloutsos, "Link positions matter: A noncommutative routing metric for wireless mesh network," *INFOCOM 2008. The 27th Conference on Computer Communications. IEEE*, pp. 744–752, April 2008.

[42] C. Koksal and H. Balakrishnan, "Quality-aware routing metrics for time-varying wireless mesh networks," *Selected Areas in Communications, IEEE Journal on*, vol. 24, pp. 1984–1994, Nov. 2006.

[43] R. Krishnan, A. Raniwala, and T. cker Chiueh, "Design of a channel characteristics-aware routing protocol," *INFOCOM 2008. The 27th Conference on Computer Communications. IEEE*, pp. 2441–2449, April 2008.

[44] H. Zhai and Y. Fang, "Impact of routing metrics on path capacity in multirate and multihop wireless ad hoc networks," in *ICNP '06: Proceedings of the Proceedings of the 2006 IEEE International Conference on Network Protocols*, (Washington, DC, USA), pp. 86–95, IEEE Computer Society, 2006.

[45] S. Roy, D. Koutsonikolas, S. Das, and Y. Hu, "High-throughput multicast routing metrics in wireless mesh networks," *Distributed Computing Systems, 2006. ICDCS 2006. 26th IEEE International Conference on*, pp. 48–48, 2006.

[46] K.-H. Kim and K. G. Shin, "On accurate measurement of link quality in multi-hop wireless mesh networks," in *MobiCom '06: Proceedings of the 12th annual international conference on Mobile computing and networking*, (New York, NY, USA), pp. 38–49, ACM, 2006.

[47] Y. Yang, J. Wang, and R. Kravets, "Designing routing metrics for mesh networks," *Wireless Mesh Networks, 2005. WiMesh 2005. 1st IEEE Workshop on*, 2005.

[48] K. Ramachandran, I. Sheriff, E. Belding, and K. Almeroth, "Routing stability in static wireless mesh networks," in *Proceedings of the eighth Passive and Active Measurement conference*, April 2007.

[49] K. N. Ramachandran, E. M. Belding, K. C. Almeroth, and M. M. Buddhikot, "Interference-aware channel assignment in multi-radio wireless mesh networks," in *INFOCOM 2006. 25th IEEE International Conference on Computer Communications. Proceedings*, pp. 1 –12, april 2006.

[50] A. Subramanian, H. Gupta, and S. Das, "Minimum interference channel assignment in multi-radio wireless mesh networks," *Sensor, Mesh and Ad Hoc Communications and Networks, 2007. SECON '07. 4th Annual IEEE Communications Society Conference on*, pp. 481–490, June 2007.

[51] P. Bahl, R. Chandra, and J. Dunagan, "Ssch: slotted seeded channel hopping for capacity improvement in ieee 802.11 ad-hoc wireless networks," in *MobiCom '04: Proceedings of the 10th annual international conference on Mobile computing and networking*, (New York, NY, USA), pp. 216–230, ACM, 2004.

[52] T. Parker and K. Langendoen, "Guesswork: robust routing in an uncertain world," *Mobile Adhoc and Sensor Systems Conference, 2005. IEEE International Conference on*, pp. 9 pp.–, Nov. 2005.

[53] S. Chachulski, M. Jennings, S. Katti, and D. Katabi, "Trading structure for randomness in wireless opportunistic routing," in *SIGCOMM '07: Proceedings of the 2007 conference on Applications, technologies, architectures, and protocols for computer communications*, (New York, NY, USA), pp. 169–180, ACM, 2007.

[54] B. Radunovic, C. Gkantsidis, P. Key, and P. Rodriguez, "An optimization framework for opportunistic multipath routing in wireless mesh networks," *INFOCOM 2008. The 27th Conference on Computer Communications. IEEE*, pp. 2252–2260, April 2008.

[55] K. Zeng, W. Lou, and H. Zhai, "On end-to-end throughput of opportunistic routing in multirate and multihop wireless networks," *INFOCOM 2008. The 27th Conference on Computer Communications. IEEE*, pp. 816–824, April 2008.

[56] M. Zorzi and R. Rao, "Geographic random forwarding (geraf) for ad hoc and sensor networks: multihop performance," *Mobile Computing, IEEE Transactions on*, vol. 2, pp. 337–348, Oct.-Dec. 2003.

[57] D. Aguayo, J. Bicket, S. Biswas, G. Judd, and R. Morris, "Link-level measurements from an 802.11b mesh network," *SIGCOMM Comput. Commun. Rev.*, vol. 34, no. 4, pp. 121–132, 2004.

[58] T. Leighton and S. Rao, "An approximate max-flow min-cut theorem for uniform multicommodity flow problems with applications to approximation algorithms," *Symposium on Foundations of Computer Science*, vol. 0, pp. 422–431, 1988.

[59] N. Garg and J. Konemann, "Faster and simpler algorithms for multicommodity flow and other fractional packing problems," in *Proc. of IEEE FOCS*, 1998.

[60] L. Fleischer, "Approximating fractional multicommodity flow independent of the number of commodities," *Foundations of Computer Science, 1999. 40th Annual Symposium on*, pp. 24–31, 1999.

[61] M. Kodialam and T. Nandagopal, "On the capacity region of multi-radio multi-channel wireless mesh networks," *Wireless Mesh Networks, 2005. WiMesh 2005. 1st IEEE Workshop on*, 2005.

[62] V. Bhandari and N. Vaidya, "Connectivity and capacity of multi-channel wireless networks with channel switching constraints," *INFOCOM 2007. 26th IEEE International Conference on Computer Communications. IEEE*, pp. 785–793, May 2007.

[63] J. Robinson, M. Uysal, R. Swaminathan, and E. Knightly, "Adding capacity points to a wireless mesh network using local search," *INFOCOM 2008. The 27th Conference on Computer Communications. IEEE*, pp. 1247–1255, April 2008.

[64] Y. Gao, D.-M. Chiu, and J. C. Lui, "Determining the end-to-end throughput capacity in multi-hop networks: methodology and applications," in *SIGMETRICS '06/Performance '06: Proceedings of the joint international conference on Measurement and modeling of computer systems*, (New York, NY, USA), pp. 39–50, ACM, 2006.

[65] P. Kyasanur and N. H. Vaidya, "Capacity of multi-channel wireless networks: impact of number of channels and interfaces," in *MobiCom '05: Proceedings of the 11th annual international conference on Mobile computing and networking*, (New York, NY, USA), pp. 43–57, ACM, 2005.

[66] D. Kotz and K. Essien, "Analysis of a campus-wide wireless network," in *ACM MobiCom*, pp. 107–118, 2002.

[67] T. Henderson, D. Kotz, and I. Abyzov, "The changing usage of a mature campus-wide wireless network," in *Proc. of ACM MobiCom*, pp. 187–201, 2004.

[68] A. Balachandran, G. M. Voelker, P. Bahl, and P. V. Rangan, "Characterizing user behavior and network performance in a public wireless lan," *SIGMETRICS Perform. Eval. Rev.*, vol. 30, no. 1, pp. 195–205, 2002.

[69] R. Jain, D. Lelescu, and M. Balakrishnan, "Model t: an empirical model for user registration patterns in a campus wireless lan," in *Proc. of MobiCom*, pp. 170–184, 2005.

[70] G. B. Dantzig, "Linear programming under uncertainty," *Manage. Sci.*, vol. 1, no. 3-4, pp. 197–206, 1955.

[71] A. Ben-Tal and A. Nemirovski, "Robust solutions of linear programming problems contaminated with uncertain data," *Mathematical Programming*, vol. 88, no. 3, pp. 411–424, 2000.

[72] L. E. Ghaoui, F. Oustry, and H. Lebret, "Robust solutions to uncertain semidefinite programs," *SIAM J. Optimization*, vol. 9, pp. 33–52, 1998.

[73] A. Atamturk and M. Zhang, “Two-stage robust network flow and design under demand uncertainty,” *OPERATIONS RESEARCH*, vol. 55, no. 4, pp. 662–673, 2007.

[74] K. Yang, Y. Wu, J. Huang, X. Wang, and S. Verdu, “Distributed robust optimization for communication networks,” *INFOCOM 2008. The 27th Conference on Computer Communications. IEEE*, pp. 1157–1165, April 2008.

[75] D. Bertsimas and M. Sim, “Robust discrete optimization and network flows,” *Mathematical Programming*, vol. 98, no. 1, pp. 49–71, 2004.

[76] J. Wellons, L. Dai, Y. Xue, and Y. Cui, “Augmenting predictive with oblivious routing for wireless mesh networks under traffic uncertainty,” *Comput. Netw.*, vol. 54, pp. 178–195, February 2010.

[77] K. Jain, J. Padhye, V. Padmanabhan, and L. Qiu, “Impact of interference on multi-hop wireless network performance,” in *Proc. of Mobicom*, September 2003.

[78] V. S. A. Kumar, M. V. Marathe, S. Parthasarathy, and A. Srinivasan, “Algorithmic aspects of capacity in wireless networks,” in *Proc. of ACM SIGMETRICS*, pp. 133–144, 2005.

[79] Y. Xue, B. Li, and K. Nahrstedt, “Optimal resource allocation in wireless ad hoc networks: A price-based approach,” *IEEE Transactions on Mobile Computing*, vol. 5, pp. 347–364, April 2006.

[80] C. E. Koksal, K. Jamieson, E. Telatar, and P. Thiran, “Impacts of channel variability on link-level throughput in wireless networks,” *SIGMETRICS Perform. Eval. Rev.*, vol. 34, no. 1, pp. 51–62, 2006.

[81] J. Camp, J. Robinson, C. Steger, and E. Knightly, “Measurement driven deployment of a two-tier urban mesh access network,” in *MobiSys ’06: Proceedings of the 4th international conference on Mobile systems, applications and services*, (New York, NY, USA), pp. 96–109, ACM, 2006.

[82] C. Reis, R. Mahajan, M. Rodrig, D. Wetherall, and J. Zahorjan, “Measurement-based models of delivery and interference in static wireless networks,” in *SIGCOMM ’06: Proceedings of the 2006 conference on Applications, technologies, architectures, and protocols for computer communications*, (New York, NY, USA), pp. 51–62, ACM, 2006.

[83] V. Gambiroza, B. Sadeghi, and E. W. Knightly, “End-to-end performance and fairness in multihop wireless backhaul networks,” in *MobiCom ’04: Proceedings of the 10th annual international conference on Mobile computing and networking*, (New York, NY, USA), pp. 287–301, ACM, 2004.

[84] A. Kashyap, S. Ganguly, and S. R. Das, "A measurement-based approach to modeling link capacity in 802.11-based wireless networks," in *MobiCom '07: Proceedings of the 13th annual ACM international conference on Mobile computing and networking*, (New York, NY, USA), pp. 242–253, ACM, 2007.

[85] D. Tang and M. Baker, "Analysis of a local-area wireless network," in *MobiCom '00: Proceedings of the 6th annual international conference on Mobile computing and networking*, (New York, NY, USA), pp. 1–10, ACM, 2000.

[86] D. Kotz and K. Essien, "Analysis of a campus-wide wireless network," *Wireless Networks*, vol. 11, no. 1, pp. 115–133, 2005.

[87] M. Balazinska and P. Castro, "Characterizing mobility and network usage in a corporate wireless local-area network," in *MobiSys '03: Proceedings of the 1st international conference on Mobile systems, applications and services*, (New York, NY, USA), pp. 303–316, ACM, 2003.

[88] D. Schwab and R. Bunt, "Characterising the use of a campus wireless network," *INFOCOM 2004. Twenty-third AnnualJoint Conference of the IEEE Computer and Communications Societies*, vol. 2, pp. 862–870 vol.2, March 2004.

[89] A. Lakhina, K. Papagiannaki, M. Crovella, C. Diot, E. D. Kolaczyk, and N. Taft, "Structural analysis of network traffic flows," in *SIGMETRICS '04/Performance '04: Proceedings of the joint international conference on Measurement and modeling of computer systems*, (New York, NY, USA), pp. 61–72, ACM, 2004.

[90] S. Karpinski, E. M. Belding, and K. C. Almeroth, "Wireless traffic: The failure of cbr modeling," *Broadband Communications, Networks and Systems, 2007. BROADNETS 2007. Fourth International Conference on*, pp. 660–669, Sept. 2007.

[91] S. Karpinski, E. Belding, and K. Almeroth, "Towards realistic models of wireless workload," *Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks and Workshops, 2007. WiOpt 2007. 5th International Symposium on*, pp. 1–5, April 2007.

[92] G. E. Box and G. M. Jenkins, *Time Series Analysis: Forecast and Control.* San Francisco, CA, USA: Holden-Day, 1970.

[93] C. Chatfield, *Time-Series Forecasting.* Chapman and Hall / CRC, 2001.

[94] W. Vandaele, *Applied time series and Box-Jenkins models.* New York, NY, USA: Addison-Wesley, 1983.

[95] W. W. Wei, *Time series analysis: Univariate and multivariate methods*. New York, NY, USA: Addison-Wesley, 1989.

[96] J. D.Hamilton, *Time Series Snalysis*. Princeton, NJ, USA: Princeton University Press, 1994.

[97] R. J. H. Spyros Makridais, Steven C. Wheelwright, *Forecasting: Methods and Applications*. New York, NY, USA: Wiley, 1998.

[98] M. Kendall and J. K. Ord, *Time Series*. London, UK: Griffin, 1990.

[99] G. Mao, "Real-time network traffic prediction based on a multiscale decomposition," pp. 492–499, 2005.

[100] "Darmouth crawdad." http://crawdad.cs.dartmouth.edu/.

[101] H. Wang, H. Xie, L. Qiu, Y. R. Yang, Y. Zhang, and A. Greenberg, "Cope: traffic engineering in dynamic networks," in *Proc. of ACM SIGCOMM*, 2006.

[102] "Ilog cplex mathematical programming optimizers." http://www.ilog.com/products/cplex.

[103] P. Kyasanur, "Routing and interface assignment in multi-channel multi-interface wireless networks," in *In IEEE WCNC*, pp. 2051–2056, 2005.

[104] A. P. Subramanian, H. Gupta, S. R. Das, and J. Cao, "Minimum interference channel assignment in multiradio wireless mesh networks," *IEEE Transactions on Mobile Computing*, vol. 7, pp. 1459–1473, 2008.