This is to certify that the
dissertation entitled

RESOURCE ALLOCATION AND ROUTING IN MULTI-HOP WIRELESS NETWORKS

presented by

Yong Ding

has been accepted towards fulfillment
of the requirements for the

Ph.D. degree in Computer Science

Major Professor's Signature

5/8/2010

Date

RESOURCE ALLOCATION AND ROUTING IN MULTI-HOP WIRELESS NETWORKS

By

Yong Ding

A DISSERTATION

Submitted to
Michigan State University
in partial fulfillment of the requirements
for the degree of

DOCTOR OF PHILOSOPHY

Computer Science

2010

ABSTRACT

RESOURCE ALLOCATION AND ROUTING IN MULTI-HOP WIRELESS NETWORKS

By

Yong Ding

Multi-hop wireless networks have been widely deployed in different application areas. These include wireless sensor networks, wireless mesh networks, and wireless vehicular ad-hoc networks. Resource allocation and routing are two critical issues for multi-hop wireless networks. The appropriate allocation of the limited resources, such as energy or channel, can usually improve the network performance dramatically. On the other hand, routing is usually coupled with resource allocation, which affects the network topology. The problem itself can even be regarded as a high level resource allocation, because it is equivalent to allocating the set of wireless links and bandwidth for each particular application. Due to the difference in hardware characteristics and application background, different networks should be given special design considerations. In this dissertation, we discuss several research topics related with resource allocation and routing in multi-hop wireless networks.

First, we investigate the problem of sleep scheduling of sensors in dense wireless sensor networks with the goal of reducing energy consumption. The basic idea is to partition sensors into groups such that a connected backbone network can be maintained by keeping only one arbitrary node from each group in active status. Unlike previous approaches that use geographic partitions, we propose to partition nodes based on their measured connectivity. The proposed scheme not only ensures K-vertex connectivity of the backbone network, but also outperforms other approaches in irregular radio environments.

Second, we study the channel assignment with partially overlapping channels in wireless mesh networks. Unlike previous studies that focus on the channel assignment with non-overlapping channels only, we propose efficient channel assignment algorithms that use both non-overlapping and overlapping channels. We show that the network capacity can be dramatically increased by using partially overlapping channels, and the proposed algorithms find better solutions than existing algorithms.

Third, as both static channel assignment and dynamic channel assignment have their own strengths and drawbacks, we propose a hybrid wireless mesh networking architecture, which combines the advantages of both channel assignment approaches. We discuss both the channel assignment algorithms and routing protocols used in this hybrid architecture. We demonstrate that our proposed approach achieves better adaptivity to the changing traffic and lower data delivery delay.

Fourth, we study the application of multi-source video on-demand streaming in wireless mesh networks. We focus on the problem of finding the maximum number of high-quality and independent paths from the user to the multiple video sources by considering the effect of wireless interference. We propose efficient routing algorithms that aim at minimizing the network congestion caused by each new video session. We show that it not only improves the average video streaming performance, but also increase the network's capacity of satisfying video requests.

Finally, we present a routing algorithm for vehicular ad-hoc networks with the goal of efficient data delivery under low and median vehicle densities. We consider the deployment of static nodes at road intersections to help relay data. We propose an algorithm that determines whether a vehicle forwards a packet to the static node and when the static node forwards the packet to vehicles. We show that our proposed routing algorithm achieves lower data delivery delay than the previous algorithms.

ACKNOWLEDGMENTS

I would like to express my sincere gratitude to Professor Li Xiao, whose encouragement, supervision and support from the preliminary to the concluding level enabled me to complete this dissertation.

I also gratefully acknowledge my dissertation committee, Professor Matt Mutka, Professor Abdol-Hossein Esfahanian, and Professor Kirk Dolan, for their valuable guidance and comments.

Many thanks to my collaborators of Chen Wang, Guokai Zeng, Kanthakumar Pongaliur, Yi Huang, Yang Yang, and Pei Huang for their help in technical discussion and experiments on various research topics.

Finally, I am grateful to my parents and my wife for supporting me all the time.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

Introduction

Multi-hop wireless networks consists of wireless nodes that form a multi-hop network topology. There have been several different types of multi-hop wireless networks that are widely used in different application areas. These include wireless sensor networks (WSN), wireless mesh networks (WMN), and vehicular ad-hoc networks (VANET). As different types of networks differ in the hardware characteristics of wireless nodes and supported applications, they require different design considerations.

WSN is composed of battery-powered sensors. After deployment, WSN is intended to work for months or years. Therefore, energy becomes a critical resource. Experiments have shown that most of the sensor's energy is consumed in wireless communication. This motivates us to study the design of energy-efficient algorithms in WSN to prolong the network lifetime.

WMN is composed of mesh routers that have permanent power supply. Unlike WSN, WMN is used as infrastructures to enhance wireless coverage and last-mile Internet access. Therefore, energy is no longer an issue in WMN. Instead, it becomes important to improve the overall network throughput. In multi-channel WMN, the appropriate assignment of channels is critical for improving the network capacity. As more and more high bit-rate applications, such as video streaming, begin to be de-

ployed on WMNs, routing becomes another important issue. It is usually correlated with channel assignment. An efficient routing algorithm not only improves the application performance, but also increases the overall network throughput of WMN, which are supposed to be shared among multiple users. This motivates us to study the channel allocation and routing in multi-channel WMN.

VANET consists of vehicles with wireless communication capability. Similar to WMN, energy is not an issue, because the wireless device can be powered by vehicles. Unlike WMN, which is static, the VANET has high mobility and the network is frequently disconnected. As a result, it usually causes high delay for end-to-end data delivery over multiple hops. This motivates us to study efficient routing algorithms to reduce the data delivery delay in VANET.

In this chapter, we first give an overview of the different types of multi-hop wireless networks. We then discuss the challenges of resource allocation and routing in different networks. Finally, we present the structure of dissertation.

1.1 Overview of Multi-Hop Wireless Networks

In this section, we present a brief introduction of different types of multi-hop wireless networks and their potential applications.

1.1.1 Wireless Sensor Networks

A wireless sensor network (WSN) is composed of a large number of tiny battery-powered sensor nodes that are densely deployed in the area of interest to monitor the environment and collect data. Each sensor node consists of sensing, data processing, and wireless communication components. Usually, sensor nodes are randomly deployed, and they are able to self-organize a multi-hop wireless network. Fig.1.1 [9] illustrates a typical architecture of a wireless sensor network. Data are collected from

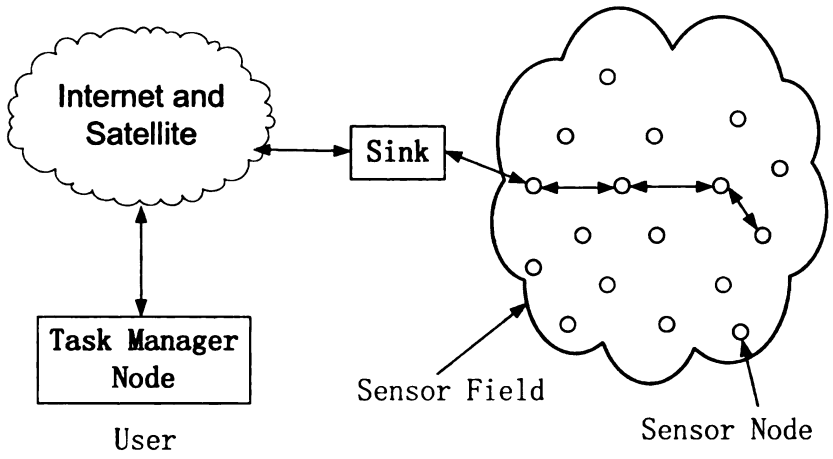


Figure 1.1: Wireless Sensor Network [9]

sensor nodes and routed back to the sink through multi-hop relay of other sensor nodes. The user communicates with the sink via Internet or Satellite to view results from and control the sensor network.

The advancement of technology in wireless sensor networks has enabled a variety of applications [9] [24].

- Military Applications: such as monitoring friendly forces, reconnaissance of opposing forces, and battlefield surveillance.
- Environmental Applications: such as forest fire detection, bio-complexity mapping of the environment, and flood detection.
- Health Applications: such as tele-monitoring of human physiological data, Tracking and monitoring patients inside a hospital, and drug administration in hospitals.
- Home and Commercial Applications: such as home automation and smart environment, managing inventory control, and vehicle tracking and detection.

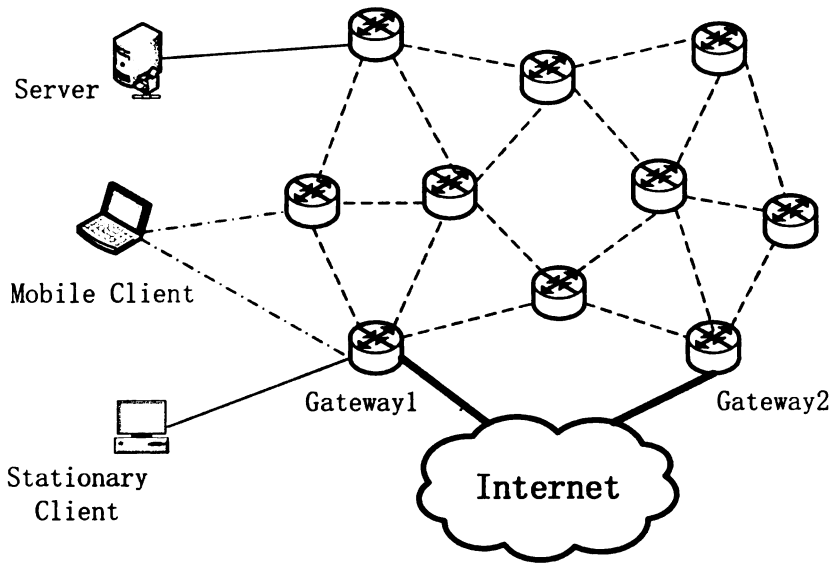


Figure 1.2: Wireless Mesh Network [7]

1.1.2 Wireless Mesh Networks

Wireless mesh networking has become a promising technology that has the potential to enable many useful applications. A typical wireless mesh network (WMN) architecture is shown in Fig.1.2 [7].

A wireless mesh network consists of mesh routers and mesh clients. Usually, mesh routers are assumed to be stationary, and each router is equipped with one or more wireless interfaces. While mesh routers provide coverage in the neighborhood, they also connect with each other to form a multi-hop wireless backbone network. Some special mesh routers, called gateways, also have interfaces directly connected to the Internet. Thus, the backbone network can carry traffic between mesh clients or between mesh clients and the Internet through multi-hop relay of the mesh routers. A mesh client can be mobile or stationary and typically has only one interface. It is connected to the network through the mesh router that covers the client or directly through the gateway.

In a wireless mesh network, there are three kinds of links: i) Access links, which connect users to mesh routers. They can be wired (Ethernet) or wireless (802.11 or Bluetooth); ii) Backbone links, which provide connections between mesh routers. Usually 802.11 or some proprietary protocols are used. iii) Backhaul links, which connect gateways to the Internet. The technologies usually used are Ethernet or TV cable for wired connections, and 802.16 or proprietary protocols for wireless connections.

There have been many prospective applications envisioned by wireless mesh networking [10]. Some commercial deployments and university testbeds are already working with the focus on different application areas.

- Extended 802.11 coverage: Broadband home networking is a typical application of using wireless mesh networking technology to extend 802.11 indoor coverage. Traditional 802.11 WLAN may either cause dead zones in the home if only one access point is deployed, or cause unnecessary traffic under the deployment of multiple access points because end users of different access points have to communicate through the Internet even though they are in the same home. By placing multiple wireless mesh routers with mesh connectivity in the home, all these problems can be solved.

- Community Networks: A community network can be formed by establishing mesh connectivity among mesh routers in the homes within a community area. This brings two major advantages. First, multiple homes may share one access point that connects to the Internet, which is a more flexible and economic way to access the Internet. Second, wireless mesh community networks enable many applications within the neighborhood, such as distributed file storage, file access and video streaming.

- Broadband Internet Access: Wireless mesh networking has also been considered as a cost-efficient way of establishing last-mile Internet access. By replacing

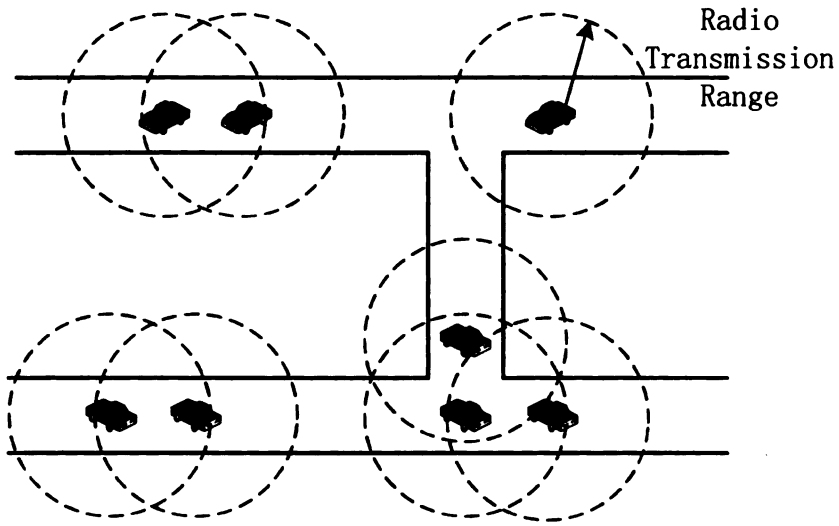


Figure 1.3: Wireless Vehicular Ad-Hoc Network

wired connections with wireless connections using mesh routers, not only can the coverage be enhanced, but the network also becomes more flexible.

- Other Applications: These include transportation systems, building automation, health and medical systems, and security surveillance systems.

1.1.3 Wireless Vehicular Ad-Hoc Networks

Since FCC allocated 75 MHz of spectrum at 5.9GHz for Dedicated Short-Range Communications (DSRC) [1], inter-vehicle communication and vehicle-to-infrastructure communication can be supported using a variant of the IEEE 802.11a technology. Vehicular ad-hoc network (VANET) is composed of vehicles, where each vehicle is equipped with wireless communication capability. An example of vehicular network is illustrated in Fig.1.3. The data can be delivered between vehicles through multi-hop data transmission.

Many potentially useful applications have been envisioned in vehicular ad-hoc networks [115] [119].

- Safety Applications: such as vehicle collision avoidance and road condition warning.
- Road Assistance Applications: such as real-time traffic estimation for trip planning, and information retrieval.
- Entertainment Application: such as media content sharing.
- Data Deliver Networks: For example, by embedding sensors in vehicles, a mobile sensor network can be established to monitor road and environmental conditions in large areas. The vehicular networks can act as a "delivery networks" to transfer data from remote sensor-nets to Internet servers.

1.2 Motivation

In this section, we introduce the challenges in resource allocation and routing in different types of multi-hop wireless networks, and therefore explain the motivation of this dissertation.

1.2.1 Critical Resources in Wireless Networks

In this subsection, we introduce the critical resources in wireless sensor and mesh networks. The resource allocation becomes an important issue in both networks.

1. Energy Efficiency in Wireless Sensor Networks

Wireless sensor networks consist of a large number of battery powered nodes that need to operate in unattended status for months. In order to sustain sensors to run for a long period of time with limited energy capacity, it is critical to save energy in sensor operations. Since wireless communication consumes the majority of energy among all the sensors' activities, reducing power consumption in communication is

the most effective approach to prolong sensors' lifetime. Two strategies are usually used to minimize energy dissipation in sensor communication: i) adjust the radio transmission power of each node; or ii) schedule the wireless interfaces of sensor nodes to rotate between active and sleeping status.

When a wireless interface is turned on (or in active status), it may be at one of the three modes; transmitting packets, receiving packets, and idle listening. Experiments in [20] showed that the energy consumption ratio of listening, receiving, and transmitting is 1:1.2:1.7 for sensor nodes. As a result, if an active interface is not transmitting or receiving packets frequently, it may consume a large portion of energy on idle listening. In this dissertation, we propose an adaptive sleep scheduling algorithm [33] [35] that enables sensor nodes to save the energy spent on idle listening and therefore prolongs the network's lifetime.

2. Channel Diversity in Wireless Mesh Networks

One major problem facing wireless networks is the capacity reduction caused by the interference among wireless links. For example, the throughput of a 2-hop flow halves in a single-radio single-channel multi-hop WMN, because the wireless interference enforces that only one of the two hops can be active at the same time.

It has been shown that a pair of wireless links can transmit simultaneously as long as they are operating on different non-overlapping channels, even through they are within interference range of each other. Fortunately, 802.11b/g and 802.11a provide 3 and 12 non-overlapping channels respectively. Thus, we can mitigate the interference within the network by fully exploiting the spectrum resource.

Although multiple channels and multiple radios are available, standard IEEE 802.11 protocol is designed for only single channel and single radio per node. In a multi-channel multi-radio WMN, a coordination layer is needed between the MAC and network layer to enable a mesh node to coordinate among multiple radios and

configure them with appropriate channels. Channel allocation is therefore a very important issue. Generally speaking, we want to assign each radio of each mesh node with an appropriate channel so that the network performance can be maximized.

In this dissertation, we propose an efficient channel assignment algorithm [31] that fully utilizes both partially overlapping channels and non-overlapping channels in the channel assignment. In addition, we propose a hybrid wireless mesh networking architecture [32], which well utilizes the advantages of both static and dynamic channel allocation mechanisms.

1.2.2 Routing in Multi-Hop Wireless Networks

Due to the different characteristics of different types of multi-hop wireless networks, the routing protocols have different design considerations in different networks.

1. Routing in Multi-Channel Wireless Mesh Networks

Wireless mesh network is envisioned to be used for low-cost infrastructures for building community networks. A community network is a static multi-hop wireless network composed of many mesh routers, where each mesh router establishes connectivity with neighboring mesh routers. There are some special routers working as gateways to provide access to the Internet.

One distinguished feature of wireless mesh networks is that multiple channels are usually used to improve the network capacity. Therefore, there is one more important factor that we need to consider in the design of routing protocols, that is, the channel diversity. When finding the route for a single flow, we need to ensure that the route has good channel diversity so that the intra-flow interference is minimized. While considering the routing of multiple flows, in addition to the intra-flow interference of each flow, we also need to minimize the inter-flow interference between different flows, so that we can maximize the overall performance.

As the use of multiple channels improves the capacity of wireless mesh network, there is growing demand of deploying high performance applications, such as video on-demand applications, on the network. To improve the video streaming performance, we can stream the video from multiple paths over the wireless mesh network. The multipath video streaming brings the following benefits compared to the single-path video streaming: (1) The multipath routing distributes the traffic over the network, so it enjoys higher quality of data transmission than single-path routing, which can easily cause congestion; (2) The multipath routing improves the stability and robustness, because the probability that all paths experience bursty loss or congestion gets smaller than using single path. In this dissertation, we propose a multipath routing algorithm for improving the multi-source video streaming performance in multi-channel wireless mesh networks.

2. Routing in Vehicular Ad-Hoc Networks

A vehicular ad-hoc network can be regarded as a special type of mobile ad hoc network (MANET) with some unique features. First, as vehicles usually move at high speeds, the topology of the vehicular network changes more rapidly. Second, unlike MANETs where an end-to-end connection is usually assumed, vehicular networks are frequently disconnected depending on the vehicle density. The most distinguished feature is that vehicles have some restricted mobility pattern. Specifically, all vehicle movements are constrained in roads, which have a static structure. Vehicles can only move in either direction on the road, and move at a speed restricted by the speed limit. These characteristics make the classical MANET routing algorithms such as AODV [83] and GPSR [55] inefficient in vehicular networks, and significantly influence the design of alternative routing protocols.

As the vehicular network is frequently disconnected, there might be cases when a packet has no connected vehicle to deliver to as the next hop. In this case, vehi-

cles have to carry the packet and then forward it when possible. As a result, the data delivery delay may become very large. Especially, the vehicular network has lots of topology holes, which makes traditional greedy geographic forwarding every inefficient. In this case, we can utilize the mobility pattern of vehicles to reduce the packet delivery delay. Although the vehicles have high mobility, the street map is static. In this dissertation, we propose a static-node assisted routing protocol [34] that well utilizes the knowledge of street map to achieve improved data delivery delay in vehicular ad-hoc networks.

1.3 Structure of The Content

The remainder of this dissertation is organized as follows. We review the related work in Chapter 2. Chapter 3 presents a distributed sleeping scheduling algorithm in wireless sensor networks, which is adaptive to irregular radio environments. Chapter 4 proposes efficient channel assignment algorithms with partially overlapping channels in wireless mesh networks. We discuss the hybrid channel allocation for wireless mesh networks in Chapter 5. In Chapter 6, we propose an efficient multipath routing algorithm, which improves the video streaming performance in wireless mesh networks. In Chapter 7, we present a static-node assisted routing protocol for vehicular ad-hoc networks, which reduces the data delivery delay. Finally, we present a summary and some potential future work in Chapter 8.

C

R

In t

dif

2.1

En

ea

re

ta

a c

to

su

in

no

in

ne

pa

in

CHAPTER 2

Related Work

In this chapter, we review the previous work on resource allocation and routing in different types of multi-hop wireless networks.

2.1 Energy Efficient Wireless Sensor Networks

Energy in sensor networks can be saved by adjusting the radio transmit power of each node. Several topology control algorithms [89] [109] [63] have been proposed to reduce energy consumption by selecting adequate node transmit power while maintaining network connectivity. Ramanathan and Rosales-Hain [89] formulated it as a constrained optimization problem and presented distributed heuristic algorithms to maintain a connected topology using minimum power. Wattenhofer et al. [109] suggested to decide the radio transmit power of each node based on the directional information, that is, a node grows its transmission power until it finds a neighbor node in every direction. Li and Hou [63] proposed $FLSS_k$, which minimizes the maximum transmission power used while preserving k-vertex connectivity of the wireless network. In addition, some other work studied energy-saving specifically in routing packets, such as energy conserving routing [18] and efficient communication proposed in [65]. Energy conserving routing [18] selects the appropriate routes and correspond-

ing power levels in order to maximize the network lifetime. Li and Song [65] proposed a topology control algorithm for each node to adjust its transmission power, such that the topology formed is efficient for both unicast and broadcast communications.

In [25] [45] [13], the authors select the set of active nodes for routing purposes based on the idea of approximating a minimum connected dominating set (MCDS). [112] further discusses how to balance energy dissipation in the cluster heads of the CDS. Deb and Nath [27] proposed a node scheduling approach that can adapt to the trade-off between energy conservation and data delivery quality. Although CDS approaches save energy by decreasing the number of active nodes, they are not efficient at balancing energy consumption among nodes so as to maximize the network lifetime.

To reduce the energy waste in idle listening, duty cycling has been proposed in [118] and [106], where the wireless interface of each node follows a periodic cycle of active/sleep states. Although duty cycling is energy-efficient, it increases the delay of data delivery, because the intermediate node has to wait for the next-hop node to wake up to receive the packet. [36] analyzes the bounds of data delivery delay by using completely decentralized duty cycling. In [42], the authors formulate the problem of assigning duty cycle to each node while minimizing the end-to-end communication delay, and provide optimal solutions for networks with some special topologies and heuristic solutions for networks with arbitrary topologies.

Node scheduling algorithms that maintain a connected dominating set and balance energy usage by switching node status have been studied in [20] and [117]. These approaches cope with the idle listening problem without causing dramatical data delivery delay. Span [20] aims at reducing energy consumption of a wireless network without significantly diminishing its capacity or connectivity. In Span, each node makes a local decision on whether to sleep or join the backbone as a coordinator by periodically checking the status of its neighbors. Unlike Span, GAF [117] divides nodes into groups such that a communication backbone is formed by selecting an arbi-

trary active node from each group while keeping others in sleeping mode. Compared with Span, GAF imposes less overhead on switching node status, because only nodes within each group need to communicate with each other for load balance purposes.

CEC [116] divides nodes into clusters based on measured connectivity similarly, but it cannot efficiently switch node status within each group like GAF. Instead, it needs to re-form clusters to balance energy consumption among nodes. Friedman [41] proposed timed grid routing, which is based on virtual grids and synchronized clocks. It aims at avoiding message collision as well as conserving energy through node scheduling. The authors of [102] focused on energy efficiency and preserving network coverage at the same time.

2.2 Channel Allocation in Multi-Channel Wireless Mesh Networks

One major problem facing WMNs is the capacity reduction caused by the interference among wireless links. When two nearby wireless links communicate on the same frequency band, they cannot transmit data simultaneously. As a result, the backbone links are usually the bottleneck of the network. To alleviate this problem, in many WMNs, mesh routers are equipped with multiple radios, which can be configured to operate on different channels. Thus, simultaneous transmission and reception are made possible.

Fig.2.1 illustrates the categorization of different channel assignment algorithms. As omni-directional antenna and directional antenna have different interference models, channel allocation should be performed differently. In addition, with regard to how frequently the channel of each radio changes, channel assignment can be categorized into static assignment algorithms and dynamic assignment algorithms. Note that there are no dynamic channel assignment algorithms for directional antenna

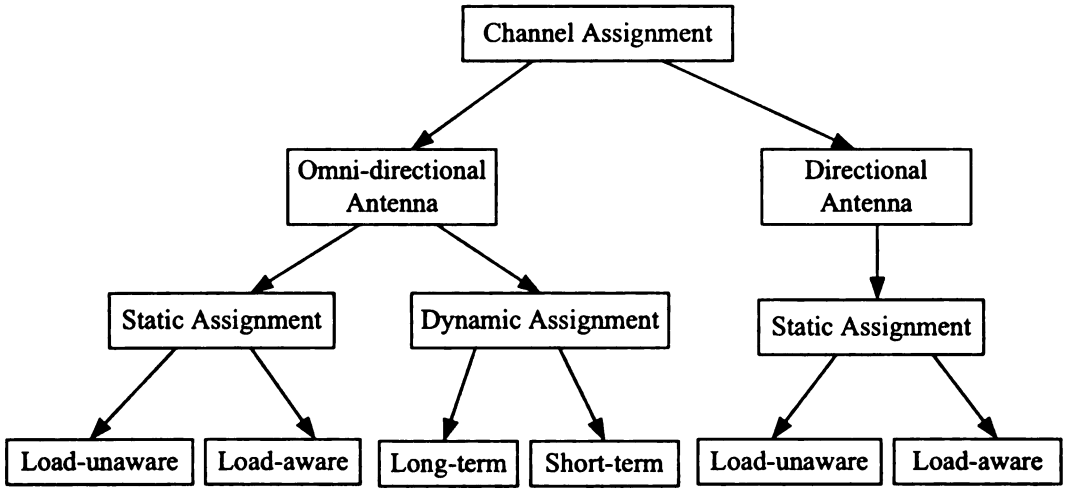


Figure 2.1: Categorization of Channel Assignment Algorithms

WMNs due to its large channel switching overhead.

In static channel allocation, one approach assumes a known traffic profile in the network, because the aggregate traffic load of each mesh router changes infrequently. The authors of [91] proposed an iterative approach to solve the joint routing and channel assignment problem, which can calculate a routing scheme as well as a channel assignment scheme, such that all traffic profiles are satisfied. In [11] and [56], the authors formulated the problem using linear programming with constraints on interference and fairness, which is NP hard. They proposed approximation algorithms to get a joint routing and channel assignment scheme.

Other studies assume that the traffic profile of each mesh router is not known, and usually consider channel assignment and routing separately. The authors of [90] assumed that the traffic from the Internet gateway to clients is dominant, and thus they first constructed a load-balanced routing tree from the original network topology, and then proposed a distributed load-aware algorithm to assign channels to the links on the tree. In [50], the peer-to-peer traffic was assumed to be dominant in the network. The authors first constructed a k-connected backbone from the original

network topology, and then assigned channels on the constructed topology. There have also been some heuristic channel assignment algorithms proposed in [98] [86] to minimize the interference in the wireless mesh network when the backbone topology is already determined.

With regard to how frequently interfaces switch channels, dynamic channel assignment can be divided into long-term and short-term strategies. In long-term strategies such as [86], channel switching is usually network-wide and governed by a central server. The central server monitors for environmental changes, recalculates the channel assignment for the whole network, and informs nodes to switch channels in each time period.

In short-term strategies such as [113] [96] [12] [101], channel switching happens frequently among nodes that want to communicate with each other. Thus, the switching overhead and coordination mechanism are the major challenges. Short-term dynamic channel assignment strategies can be categorized into two approaches. i) Devoting a single interface from each node for the purpose of control only [113]. This approach does not require synchronization among nodes but may not exploit the resources efficiently. ii) No separate interface is devoted for control so that resources can be utilized more efficiently, but this approach requires synchronization among nodes [96] [12] [101].

Some dynamic channel allocation algorithms, which require less frequent switch of channels, have been proposed in [81] [30]. The authors of [81] proposed a distributed routing and channel allocation algorithm for each flow. However, it is especially designed for the network, where each node has exactly two radios. A learning approach has been proposed in [30], where nodes autonomously learn their channel allocation based on the channel usage information in neighborhood. The learning algorithm needs time to converge to a good channel assignment each time the traffic pattern changes.

The throughput of WMNs can be improved by using directional antennas. In [88], the authors proposed using directional antennas to establish point-to-point links. They observed that even in the presence of side-lobes, it is possible to transmit (receive) along all links of a node simultaneously under the same channel. Some algorithms have been proposed in multi-radio multi-channel WMNs using direction antennas such as [87] and [38], which well utilize the special properties of directional antennae. All these algorithms use static assignment strategies, because dynamic channel switching causes much more overhead in directional antennae.

Previous channel assignment algorithms are based on non-overlapping channels. The benefit of using partially overlapping channels in WLANs has been studied in [73], [72]. In [73], the authors measured the interference between different APs when partially overlapping channels are used. They proposed channel assignment algorithms with partially overlapping channels for APs in [72], which aim at minimizing the interference between different WLANs. A similar problem has also been studied in cellular networks [122]. The authors addressed the channel assignment problem of minimizing interference between same channels and adjacent channels.

The channel assignment with partially overlapping channels in wireless mesh networks has been discussed in [74]. They assume that the traffic profile is known before doing channel assignment. The use of genetic algorithm for channel assignment has been studied in cellular networks [120], but it has not been fully studied in wireless mesh networks yet.

2.3 Routing in Multi-Hop Wireless Networks

In this section, we review the previous work on the routing in both wireless mesh networks and vehicular ad-hoc networks.

2.3.1 Multipath Routing in Wireless Mesh Networks

Video streaming is a challenging task due to its high bit rate, delay, and loss sensitivity. It is believed that the application performance can be improved by streaming the video over multiple paths. In [104] and [77], a peer-to-peer architecture for media streaming has been proposed, where a user can stream videos from both servers and peers. One major problem with multi-source video streaming is the discovery of multiple routes. Specifically, these routes need to be independent so that the failure or congestion of one route will not influence the others.

Multi-path routing has been used in wireless ad hoc networks to provide error resilience and load distribution. A number of multi-path routing protocols [53] [61] [70] have been proposed to discover multiple paths between a single source and a single destination. These protocols focus on finding edge-disjoint paths. However, there still exists correlation between these edge-disjoint paths due to the effect of wireless interference.

There have been several studies of multi-path video streaming in wireless ad hoc networks. In [69], the authors tried to find optimal paths for each session of multiple concurrent video sessions such that the overall video distortion is minimized. However, the authors have not considered wireless interference in the problem formulation. In [68] and [110], the authors took the wireless interference into consideration, and they studied the selection of optimal paths between a single source and a single destination. Both studies used Multiple Description Coded video (MDC) for multi-path streaming, but optimized for different objectives. The former one aims at minimizing the distortion metric [23], while the latter one tries to minimize concurrent packet drop probability over all paths.

Multi-path video streaming over static wireless mesh networks has been studied in [62] [57]. The authors used double-description coding and tried to find two paths to optimize the distortion of the video streaming application. However, the proposed

mechanisms are for single-channel wireless mesh networks and do not guarantee the independency among the two paths. As a result, the failure or congestion of one path may affect the other. Moreover, their optimization is only for a special case, in which the video stream is split over two paths.

A similar problem has been studied in other networks. In [14], the authors studied multi-path selection in Internet overlay networks with the goal of maximizing average video quality. In [75], the authors proposed to build a tree to aggregate videos from multiple video surveillance nodes (leaf nodes) to a center (root node) in sensor networks, while minimizing the wireless interference in the tree.

Once the multiple paths have been discovered, the sending rate on each path should also be determined. There have been many studies on rate allocation for a single video streaming session, assuming the multiple routes have already been found. They have proposed rate allocation schemes to minimize the packet loss [78], or to be used with FEC [76], or minimize the distortion metric [126]. All these rate allocation algorithms optimize for the current session only, and have not considered its influence on either existing sessions or subsequent sessions.

2.3.2 Routing in Vehicular Ad-Hoc Networks

There have been many studies on delivering messages in sparse mobile ad hoc networks with sporadic connectivity and ever-present network partitions. In these networks with extreme conditions, traditional routing algorithms that assume the fully connected path between hosts become inefficient. In case of the random movement of mobile nodes, "opportunistic forwarding" has been proposed in [105] [22] [54]. Some other studies employed controlled mobility to help message delivery [64] [124] [19].

Vehicular network (VANET) is a special type of mobile ad hoc network (MANET). The most distinguished feature of VANET is that vehicles have some restricted mobility pattern. Specifically, all vehicle movements are constrained in roads, which have

a static structure. Vehicles can only move in either direction on the road, and move at a speed restricted by the speed limit.

GPCR [66] is a position-based routing protocol, which uses the knowledge of the street map structure. In GPCR, when the packet is in a vehicle that is in the street, it will be forwarded to the next intersection by using greedy geographic forwarding. Once the packet reaches the intersection, the routing decision is made to forward the packet to the neighboring vehicle with the largest progress towards the destination. GyTAR [52] uses the vehicle density as well as the distance from the destination for route decision at intersections. When a packet reaches the intersection, it will select the neighboring intersection, which is geographically closest to the destination and has the highest vehicle traffic. "Trajectory based forwarding" [103] [79] also utilizes the static structure of street maps. It can be considered as an extension to geographic forwarding in that messages are forwarded greedily along a pre-defined trajectory.

MDDV [111] and VADD [123] combine geographic forwarding, opportunistic forwarding, and trajectory based forwarding for data delivery in VANET. They abstract each road as a link whose delay is the time consumed to deliver a packet through the corresponding road by the multi-hop communication and the carrying of moving vehicles on this road. Therefore, packets will be delivered along the shortest-delay trajectory. In cases when no vehicles are available in the next road for data delivery along the shortest delay trajectory, VADD improves packet delivery reliability by making a routing decision at each intersection to select a best currently available trajectory. Different from MDDV and VADD, our work focuses on data delivery in VANET under low vehicle densities, where VADD experiences dramatic performance degradation in packet delivery delay, and MDDV even renders poor reliability.

LOUVRE [60] abstracts the street map into an overlay network based on the vehicles at intersections and the vehicle density in each road. There exists a link if and only if the vehicle density on the road is higher than the predefined threshold.

TOPO [108] used the similar idea of overlay routing. However, they are not suitable for scenarios of low vehicle densities, because the overlay network may be disconnected most of the time. Throwbox [125], a device that can store and forward data, has been designed to improve the network throughput and reduce packet delay in delay tolerant networks. The placement of throwboxes and the determination of packet forwarding trajectory are based on the contact opportunities between each pair of mobile nodes, which can be effective in small scale networks. However, the assumption of known contact opportunities limits it extensibility to larger scale networks.

As some vehicles, such as buses, have pre-determined routes, several algorithms have been proposed to utilize this predictable mobility for data delivery in VANET. The authors of [26] suggest to utilize the non-randomness in the movement of mobile nodes. Vehicles can learn the movement pattern in the form of the meeting likelihood of pairwise nodes from the network, and use it to inform an adaptive routing strategy. The authors of [17] [16] used a similar idea and proposed routing protocols in the VANET formed by buses, whose mobility pattern can be well used to facilitate successful data delivery.

CHAPTER 3

Adaptive Sleep Scheduling for Wireless Sensor Networks

The major portion of energy in a sensor network is often consumed by idle listening instead of packet transmission and reception under light traffic or in a dense network. It has been observed that the energy consumption of a wireless interface cannot be ignored even when it is in the listening mode. The experiment in [20] shows that the energy consumption ratio of listening, receiving, and transmitting is 1:1.2:1.7. (The ratio is shown to be 1:1.05:1.4 in [97] and 1:2:2.5 in [5].) Therefore, energy can be further saved by reducing the time spent in idle listening of sensor nodes.

In duty cycling approaches [118] [106], the wireless interface of each sensor node follows a periodic cycle of active/sleep states. However, this approach incurs additional end-to-end communication delay, because the intermediate node has to wait for the node at the next hop to wake up for receiving the packet.

In this chapter, we adopt another sleep scheduling approach to reduce the energy consumption without causing dramatic data delivery delay in a dense sensor network. Since only a small portion of the sensors are involved in packet transmission and reception in a dense sensor network where broadcast is not frequently initiated, it

will be most effective to save energy by turning off the wireless interfaces of those redundant sensors that only operate in listening status. Therefore, we can divide the sensor nodes into groups such that nodes in each group are equivalent with regard to data delivery. At each time, one node is selected from each group to operate in active radio mode (listening, transmitting, receiving), while other nodes put themselves into sleeping mode by turning off their wireless interfaces. No matter which node is selected from each group, all the active nodes need to form a connected backbone network. If a sleeping node wants to send data, it can turn on its wireless interface temporarily to transmit the packets through the backbone network. In addition, the roles of active nodes and sleeping nodes need to be swapped once a while to balance the power consumption among all the nodes, which prolongs the network's lifetime.

The algorithm that saves energy by utilizing this node scheduling method has been proposed in ad hoc networks. GAF [117] [116] partitions the nodes based on their geographic locations. It divides the deployed area into multiple equal-size squared cells so that nodes in the same cell form a group. By assuming an ideal radio propagation model and choosing the appropriate side length of cells, it ensures that a well connected backbone network can be formed as long as at least one node in each cell remains in active mode. However, this geographic partition suffers from several drawbacks besides its dependence on the localization infrastructure. First, as GAF uses fixed cells in its partition, it lacks the flexibility to provide different partitions that can ensure different connectivity levels of the backbone networks. Second, geographic partition methods depend on the assumption of ideal radio propagation model, which does not always hold due to radio's irregular transmission pattern and multi-path effect in real environment. As a result, the connectivity of the backbone network cannot be guaranteed, which causes degradation in network performance.

Motivated by these limitations, we propose a Connectivity based Partition Approach (CPA), which divides nodes based on their measured connectivity instead

of guessing connectivity by their positions. In comparison, our approach has more flexibility because it can generate partitions while ensuring K-connectivity of the backbone network. In addition, as CPA is based on measured connectivity, it can guarantee the connectivity of the backbone network even under un-ideal radio propagation models. Thus, CPA is more adaptive to irregular radio environments.

The rest of the chapter is organized as follows. Section 3.1 introduces the motivation. Section 3.2 gives a formal description of the problem. Section 3.3 discusses the detailed design of CPA. Section 3.4 evaluates CPA by comparing it with GAF. Finally, we conclude this chapter in Section 3.5

3.1 GAF under Irregular Radio Model

GAF [117] identifies redundant nodes based on location information and virtual grids. It assumes an ideal radio propagation model of circular transmission range and the same radio transmission radius for all the nodes. Based on this assumption, it divides the deployed area into virtual grids with a side length of $R/\sqrt{5}$, where R is the radio transmission radius, so that each node associates itself with a corresponding grid according to its location. As any two nodes in neighboring grids are guaranteed to be connected because their distance is within R, a connected communication backbone can be formed by selecting only one active node from each grid. GAF provides a partition solution for node scheduling in a large sensor network; nevertheless, it still suffers from several limitations.

Although turning off nodes can reduce the energy consumption dramatically, the change of network graph property may affect the communication performance and therefore incur more power dissipation. Vertex connectivity is a useful metric to evaluate the communication quality of the backbone network with regard to node failure and congestion. In a K-connected network, the failure of any $K-1$ nodes

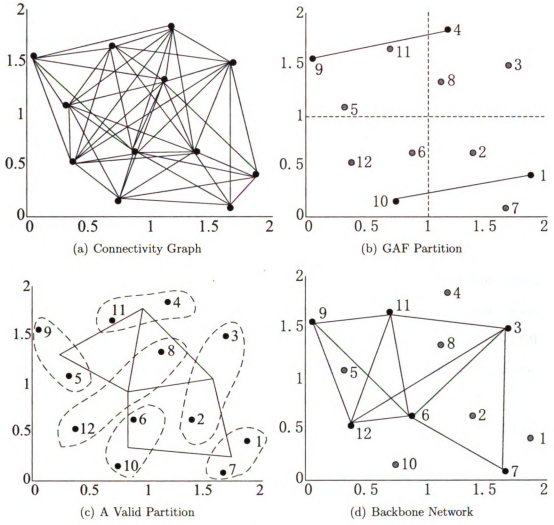


Figure 3.1: GAF under Irregular Radio Propagation Models

will not disrupt it into a disconnected graph. In addition, a network graph with higher vertex connectivity has lower possibility to have bottleneck nodes of congestion because there are at least K vertex disjoint paths between any two vertices in a K-connected graph. In the extreme case, a 1-connected graph may have cut vertices that are very likely to become congestion nodes.

GAF actually provides a 4-connected backbone network for a large sensor network. As any two nodes in neighboring grids are guaranteed to be connected, each node is connected with at least four nodes in its four neighboring grids respectively

in the backbone network. However, GAF lacks the flexibility to provide backbone networks with different vertex connectivity under different requirements. If the nodes are relatively more robust and the traffic rate is not high, a backbone network with lower connectivity is desired to achieve more energy saving by maintaining fewer active nodes. On the other hand, a backbone network with higher connectivity can cope with higher node failure and traffic rate. Therefore, a more flexible algorithm is desired to partition the nodes into groups of appropriate size.

Another problem with GAF is that it may not work well under irregular radio propagation models. To illustrate this, we use DOI (Degree of Irregularity) [46] as a radio propagation model. This model assumes an upper and lower bound on signal propagation range. The parameter DOI is defined as the maximum radio range variation per unit degree change in the direction of radio propagation. For the DOI model used in our analysis, the upper bound is the maximum radio transmission radius R, the lower bound is half of the upper bound, and DOI is set to 0.1.

We deploy 12 sensor nodes with maximum radio transmission radius of $\sqrt{5}$ uniformly into a 2×2 area as shown in Fig. 3.1(a). Each edge between pairwise sensors represents a symmetric link between them based on the DOI model. In Fig. 3.1(b), the deployed area is divided into 2×2 grids, each of which owns 3 nodes according to GAF. At each period, one node in each grid turns into active status to form a communication backbone. However, there is a possibility that the backbone graph is disconnected. As the case shown in Fig. 3.1(b), nodes 1, 4, 9, and 10 are selected from each grid to become active nodes, but they form a disconnected backbone network. The partition of GAF is invalid because the connectivity between nodes in neighboring grids is no longer ensured under the irregular radio model. They may not be able to communicate with each other even though they are within distance of R. In addition, although the connectivity can be guaranteed by halving the grid's side length so that each two nodes in neighboring grids are within distance of $R/2$,

the lower bound of radio propagation range, it is still not a good partition because it makes 4×4 grids, which is even more than the number of nodes. On the other hand, Fig. 3.1(c) shows a valid partition. It consists of 6 groups where the nodes in each group are mutually connected. Each edge between two groups means that any node from one group is connected with any node in the other group. Fig. 3.1(d) shows an example of the backbone network constructed from the partition. It is obvious that the backbone formed by selecting an arbitrary node from each group is always connected, because the graph in Fig. 3.1(c) is a connected graph. This partition is valid because it is based on the measured connectivity between nodes instead of guessing connectivity from nodes' locations. Therefore, it can ensure the connectivity between neighboring groups in the partition.

In this chapter, we propose a connectivity based partition approach (CPA), which is based on the measured connectivity between nodes. CPA aims at partitioning the nodes into groups of appropriate size to ensure K-connectivity of the backbone network, and at the adaptability in complex environments with irregular radio transmission models. Before presenting our design of CPA, a formal problem description is given in the following.

3.2 Constrained Min-Size M-Partition Problem

To reduce the energy consumption of communication in sensor networks, we can divide sensor nodes into groups such that only one node in each group keeps active at each snapshot while others are put into sleeping mode. The partition must satisfy the following constraints:

- Any node is within one hop away from all the other nodes in the same group. Under such a constraint, each node can be covered by the communication backbone, that is, each node is either in the backbone network if it is an active node

or directly connected to the backbone network if it is a sleeping node. This also enables efficient communication among nodes in each group to switch between active and sleeping modes for load-balance purposes, because each node can communicate directly with any other node in the same group.

- The backbone network formed by active nodes at each snapshot must satisfy some connectivity properties such that it does not suffer significant loss of communication quality as compared with the original network.

- The analysis in [100] shows that for those sensor applications where data are collected by a sink, the sensors closer to the sink always deplete their energy faster under uniform distribution of nodes, no matter what sleep scheduling is used. However, some mobility assisted approaches, such as [67] and [107], can help achieve uniform energy consumption in sensor networks. Therefore, in order to better evaluate the sleep scheduling algorithm, we assume the uniform energy consumption for sensor nodes in this chapter. In order for all the groups to remain alive together as long as possible, the energy needs to be evenly distributed among groups. This is because if there is a considerable number of groups with dramatically less total energy than the others, the connectivity of the backbone network will deteriorate with the early death of these lower energy groups.

- A smaller number of groups is preferred without degrading the communication quality of the original network, because more energy conservation can be achieved by decreasing the number of active nodes at each time.

By referring to some terms in graph partition problems [40], we can formalize the problem as below.

Let $G(V, E)$ be an undirected graph for the original sensor network, where each vertex in V corresponds to a sensor node and each edge in E represents a symmetric

communication link between the two nodes.

Definition 1: Given a graph $G(V, E)$, we can partition V into N disjoint sets A_1, A_2, ..., A_N, such that the induced graph of each vertex set, denoted by $G[A_i]$ ($i \in \{1, 2, ..., N\}$), is a clique. We can encode this partition by a symmetric N-by-N matrix M, where

- $M_{i,i} = 1$ ($i \in \{1, 2, ..., N\}$), representing that each $G[A_i]$ is a clique.
- For off-diagonal entries $M_{i,j}$ ($i, j \in \{1, 2, ..., N\}$, $i \neq j$)
  - $M_{i,j} = 2$ if A_i and A_j are completely adjacent, that is, any vertex in A_i is connected with any vertex in A_j.
  - $M_{i,j} = 1$ if A_i and A_j have arbitrary connections, that is, there exists some vertex in A_i connected with some vertex in A_j, but A_i and A_j are not completely adjacent.
  - $M_{i,j} = 0$ if A_i and A_j are completely non-adjacent, that is, no vertex in A_i is connected with any vertex in A_j.

This partition is called an ***M-partition*** of G, and N is the size of the partition.

Definition 2: Given an M-partition $P = \{A_1, A_2, ..., A_N\}$ of $G(V, E)$, we select an arbitrary vertex x_i from A_i ($i = 1, 2, ..., N$). Let $X = \{x_1, x_2, ..., x_N\}$. The induced graph $G[X]$ is called a ***backbone graph*** of G under M-partition P. In the rest of the chapter, we also call it *backbone network* without confusion.

Note that the M-partition implies some connectivity property of the backbone graph. Let B denote a backbone graph of G under an M-partition $P = \{A_1, A_2, ..., A_N\}$. If $M_{i,j}$ is 2, then x_i and x_j (the two vertices selected from A_i and A_j respectively) are guaranteed to be connected; if $M_{i,j}$ is 0, they must be disconnected; otherwise, they may or may not be connected.

Definition 3: Given an M-partition $P = \{A_1, A_2, ..., A_N\}$ of G, we define a graph $H(S,T)$, where each vertex $s_i \in S$ corresponds to $A_i \in P$ $(i = 1, 2, ..., N)$, and $(s_i, s_j) \in T$ if and only if $M_{i,j} = 2$. We call H the ***2-induced graph*** of P.

We are interested in H because it reflects the minimum connectivity property of any backbone graph B of G. If s_i and s_j are connected in H, then A_i is completely adjacent with A_j. Thus, in any backbone graph B of G, x_i and x_j (two arbitrary vertices selected from A_i and A_j respectively) are guaranteed to be connected. In other words, $E(H)$ is a subset of $E(B)$. Therefore, suppose $\kappa(H) = K$, then we have $\kappa(B) \geq K$, where κ denotes the vertex connectivity of the corresponding graph.

Let l be a label on V of the original network $G(V,E)$, where $l(v)$ $(v \in V)$ is the amount of energy in the sensor node v, then the total energy of the sensor network is $E_{total} = \sum_{v \in V} l(v)$. We can also derive another label g on the M-partition $P = \{A_1, A_2, ..., A_N\}$ of G, $g(A_i) = \sum_{v \in A_i} l(v)$ for each $A_i \in P$, which represents the total energy in each group of the partition.

Problem Formulation (constrained minimum-size M-partition problem): Given a graph $G(V,E)$, which represents the original sensor network, and a label l on V, which represents the amount of energy in each sensor node, find a minimum-size M-partition P^* of G such that i) $\kappa(H) \geq K$, where H is the 2-induced graph of P^* and K is the minimum vertex connectivity required by the backbone network. ii) $(1-\delta)\frac{E_{total}}{N} \leq g(A_i) \leq (1+\delta)\frac{E_{total}}{N}$ for each $A_i \in P^*$, where N is the size of P^* and $0 \leq \delta < 1$ is the unbalanced factor.

As $G[A_i]$ $(A_i \in P^*, i = 1, 2, ..., N)$ is a clique, each node is within one hop away from all the other nodes in the same group of the partition. The connectivity property of the backbone network can be guaranteed by the first constraint of the problem, and the balanced energy distribution can be satisfied by the second constraint. Moreover, the optimization nature of this problem requires the most efficient partition with regard to energy-saving.

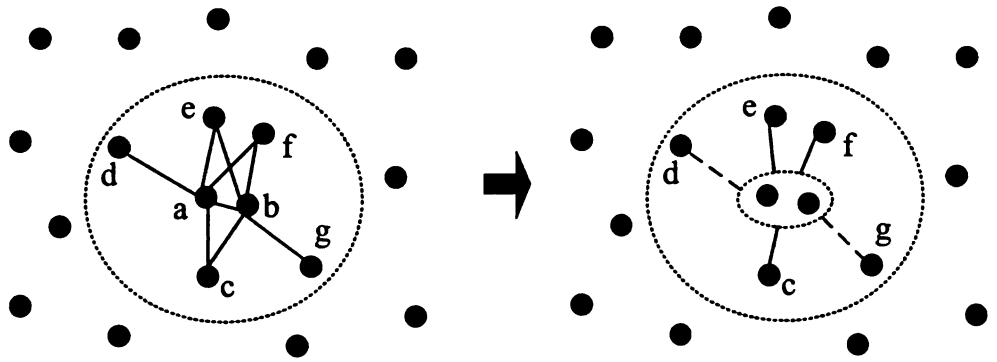
Theorem 1: The constrained minimum-size M-partition problem is NP-hard.

Proof: Consider a problem that is less hard than the formulated problem. We remove the two constraints from the formulated problem and the resulting problem becomes finding a minimum-size M-partition of G, which we denote by *smallest M-partition problem*. We then show that the *clique problem*, which is NP-Complete, can be reduced to the *smallest M-partition problem* in polynomial time. (The clique problem is the problem of determining whether a graph contains a clique of at least a given size k.) As M-partition requires that $G[A_i]$ is a clique for each $A_i \in P$, the *smallest M-partition problem* is equivalent to finding all maximal cliques in G. Suppose the set of all maximal cliques of G has been found. Then G contains a clique of size at least k if and only if there exists a maximal clique of size at least k. This reduction only needs polynomial time. Thus, the *smallest M-partition problem* is NP hard. Therefore, the constrained minimum-size M-partition problem is also NP-hard.

3.3 Connectivity based Partition Approach

In this section, we propose a Connectivity based Partition Approach (CPA) to approximate a good partition for this problem. We want to find an M-partition of which the size is as small as possible, while satisfying that any backbone graph under the M-partition is at least K-connected and the energy distribution in different groups is as even as possible. The proposed algorithm is a distributed heuristic algorithm, where only local computation is involved. CPA is a distributed iterative process. It starts from the initial partition where each node forms a unique group. CPA continuously merges two groups into a larger one until further merging will break the constraints of the problem.

In CPA, there are two kinds of nodes in each group: ordinary nodes and a head node. Each kind of node maintains its node ID and associated group information



(a) two groups a and b about to merge (b) the two groups have been merged

completely adjacent

have arbitrary connection

Figure 3.2: CPA Group Merging Process

including its group ID, IDs of other group members, and ID of the head node in its group. One head node is selected from each group to maintain some additional information on the connectivity between its group and the neighboring groups in the current M-partition. Let $N_l(A_i)$ be the set of neighboring groups that are connected with group A_i through l-value edges in the current M-partition, i.e., $N_l(A_i) = \{A_j \mid M_{i,j} = l\}$. Thus, each head node of group i will store $N_1(A_i)$ and $N_2(A_i)$, which are the set of neighboring groups having arbitrary connection with group i and completely adjacent with group i respectively.

CPA starts from the initial partition of one node in each group. Let A_i denote the group formed by node v_i. Consequently, v_i acts as the head node and stores group connectivity information $N_1(A_i)$ and $N_2(A_i)$, where $N_2(A_i)$ is the set of groups A_j whose node v_j is connected with node v_i, and $N_1(A_i)$ is empty because any two groups are either completely adjacent or completely non-adjacent in the initial partition. CPA goes through a group merging process iteratively before it gets to the final partition.

3.3.1 Group Merging

In the group merging process, head nodes of each two completely adjacent groups exchange group connectivity information to decide whether their groups should merge. Only completely adjacent groups can merge so that the new group is also a clique. Suppose A_i and A_j are two completely adjacent groups in the current M-partition. Let A_{ij} be the new group obtained by merging A_i and A_j. The group merging process first updates the group information in each node of A_{ij}, and keeps only one head node to maintain the group connectivity information in A_{ij}. Then, the new group and the neighboring groups of A_i and A_j update their group connectivity information based on the following rules. i) For each group $A' \in N_2(A_i) \cap N_2(A_j)$, A_{ij} and A' are completely adjacent (edge value of 2). ii) For each group $A' \in N_0(A_i) \cap N_0(A_j)$, A_{ij} and A' are completely non-adjacent (edge value of 0). iii) Otherwise, A_{ij} and A' have arbitrary connections (edge value of 1). Therefore, an updated M-partition is formed. Fig. 3.2 illustrates the process of merging two groups into a larger group. In Fig. 3.2(a), a and b are two completely adjacent groups, that is, any node in a is connected with any node in b. When the two groups merge as shown in Fig. 3.2(b), as groups c, e, f are completely adjacent to both a and b, each of them is completely adjacent to the new merged group. On the other hand, groups d and g only have arbitrary connection with the new group, which are illustrated by the dashed lines in the figure, because d is not completely adjacent with b and g is not completely adjacent with a.

Contentions may occur when multiple neighboring groups want to merge simultaneously. We resolve this by imposing a randomized backoff delay on the time when the two groups announce their willingness to merge. If no contention is observed at the end of the delay, the two groups about to merge will announce their decision to all of their neighboring groups. Otherwise, they will reevaluate the backoff delay based on the updates from other group merges.

The goodness of the final partition depends on the sequence of group merge. We consider several factors for deciding which two groups are preferred to be merged first in the current partition in order to arrive at a good partition eventually. These factors can be reflected as a utility function in the randomized backoff delay so that higher priority groups will announce their intentions to merge with a shorter time of delay.

- For any two groups A_i and A_j in the current partition, let $P = |N_2(A_i) \cap N_2(A_j)|$ and $Q = |N_2(A_i) \cup N_2(A_j)|$, then $C = P/Q$ indicates the level of equivalence between A_i and A_j. The two groups with higher C value will be given higher priority in the group merging process. Specifically, at the starting phase of the algorithm, where each node constitutes a single group, nodes with exactly the same set of neighbors will be merged first, because these nodes are exactly equivalent with regard to relaying data.

- Let $g(A_i)$ denote the energy in group A_i. We want the total energy to be evenly distributed in each group so as to maximize the network's lifetime. For any two groups A_i and A_j, let $D = [g(A_i) + g(A_j)]/E_{total}$ where E_{total} is the total energy of all the sensor nodes in the network, then we will give pairwise groups with lower D value higher priority in the group merging process.

Therefore, each two completely adjacent groups can be assigned with a utility value $U = k_1(1 - C) + k_2 D$, where k_1 and k_2 are coefficients. The backoff delay for each pair of groups is set to be proportional to $U + R$, where R is a random value uniformly distributed in [0, 1], which is used to resolve contentions among pairwise groups with the same utility value. As a result, the appropriate assignment of backoff delay enables pairwise groups with lower utility value to merge first as well as resolving contentions in the group merging process.

3.3.2 Guarantee of Connectivity

As we have discussed in the previous subsection, the K-connectivity of the backbone network can be guaranteed by the K-connectivity of the 2-induced graph of M-partition. However, the group merging process may disrupt this connectivity. When A_i and A_j merge, the number of completely adjacent groups for A_{ij} may decrease, and this number for each $A' \in N_2(A_i) \cup N_2(A_j)$ will decrease by 1. For example, when the groups *a* and *b* merge in Fig. 3.2, the number of completely adjacent groups for the new group decreases by 2, while the number decreases by 1 for each group of *c, d, e, f, g*. In order to ensure the K-connectivity of the 2-induced graph of M-partition, we apply a result on the property of random geometric graphs that was published in [82].

Random geometric graphs (with parameters n, r) are constructed by dropping n points randomly uniformly into the unit square (or more generally on d-dimensional Euclidean space) and adding edges to connect any two points distant at most r from each other. Therefore, a large scale sensor network with ideal radio propagation model can be modeled as a random geometric graph. Penrose [82] proved that "For a random geometric graph $G(n, r)$, let r_n, (respectively s_n) denote the minimum r at which the graph, obtained by adding an edge between each pair of points distant at most r apart, is K-connected (respectively, has minimum degree K). Then $r_n = s_n$ with probability approaching 1 as n tends to infinity." In other words, for a random geometric graph with a large number of nodes, the network reaches K-connectivity at the same time when its minimum degree reaches K.

We assume the backbone graph can be approximated as a random geometric graph, then its K-connectivity can be guaranteed by ensuring its minimum degree of K. As the 2-induced graph of M-partition is a subgraph of each backbone graph, its minimum degree is no greater than the backbone graph. Therefore, we can guarantee the K-connectivity of the backbone graph by ensuring the minimum degree of K in

the 2-induced graph of M-partition.

In M-partition, we refer *"2-degree of A_i"* to the number of groups that are completely adjacent to A_i. Each group keeps track of its 2-degree value during the group merging process, which is used to decide whether pairwise groups should be merged. If a group merge may cause the 2-degree of some group to drop below K, then these two groups will give up their intention to merge. The group merging process will be terminated when no groups can be merged.

The theoretical proof above demands the ideal radio propagation model to form a sensor network into a random geometric graph. However, our simulations in Section 3.4 show that under the irregular radio propagation model, we can still form a K-connected backbone network with high probability by ensuring the minimum degree of K in the 2-induced graph of M-partition. Compared with geographic partition methods, CPA can preserve the network's communication quality much better in irregular radio environments.

3.3.3 Centralized and Distributed Implementation

To better illustrate the idea of CPA, we first show the centralized version in Algorithm 1. During the group merging process, for each pair of completely adjacent groups, f_{ij} denotes the priority of merging these two groups, and t_{ij} indicates whether the minimum *2-degree* of the partition will drop below K (the connectivity requirement of the backbone network) if we merge them. In each step of the iteration, we will merge the two groups with the highest priority (smallest f value) that will not break the connectivity constraint (t value is $false$). The merging process continues until the merge of any pair of groups will break the constraint.

With some modifications, the algorithm can be implemented in a distributed fashion. Fig.3.3 illustrates the state transition diagram of each head node. Initially, each node constitutes a group (or is a head node). In the start, each node broadcasts an

A

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20

U

sta

adj

me

Le

co

mo

the

sta

pre

If t

into

Algorithm 1 Centralized Partitioning Algorithm

1: **for all** $v_i \in V$ **do**
2: $A_i = \{v_i\}$
3: $deg_2(A_i) =$ degree of v_i in G
4: **end for**
5: **repeat**
6: **for all** $A_i, A_j \in A$, where A_i and A_j are completely adjacent **do**
7: $C = \mid N_2(A_i) \cap N_2(A_j) \mid / \mid N_2(A_i) \cup N_2(A_j) \mid$
8: $D = [g(A_i) + g(A_j)]/E_{total}$
9: $f_{ij} = k_1(1 - C) + k_2 D$
10: $d^* = \mid N_2(A_i) \cap N_2(A_j) \mid$
11: $mindeg = \min\{d^* \cup \{deg_2(s) - 1 \mid s \in N_2(A_i) \cup N_2(A_j)\}\}$
12: $t_{ij} = (mindeg < K)\ ?\ true : false$
13: **end for**
14: Choose (A_x, A_y) with the smallest f value among all (A_i, A_j) where $t_{ij} = false$
15: **for all** $s \in N_2(A_x) \cup N_2(A_y)$ **do**
16: $deg_2(s) = deg_2(s) - 1$
17: **end for**
18: $A_x = A_x \cup A_y$, $deg_2(A_x) = \mid N_2(A_x) \cap N_2(A_y) \mid$
19: $A = A - \{A_y\}$
20: **until** $t_{ij} = true$ for all (A_i, A_j)

UPDATE_MSG to its neighbors containing its group information. In the *Decision* state, the head node calculates the f value and t value for each of its completely adjacent groups, and selects the group with the smallest f value (highest priority to merge) where t value is *false* (the merge will not break the connectivity constraint). Let A_x be the current group, and assume the best group to merge with is A_y. If any completely adjacent group of A_x or A_y is in *Merging* or *Holding* state, then the head node will go into *Waiting* state, because the merge of A_x and A_y will collide with the merge of some other groups. Otherwise, the head node will go into *Contending* state. In this state, the head node will send a *MERGE_REQ* message to A_y, expressing its willingness to merge with A_y, with a backoff delay as a function of f. If the contention succeeds (A_x receives a *MERGE_ACK* message from A_y), it goes into *Merging* state, and otherwise (A_x receives a *MERGE_NAK* message from A_y)

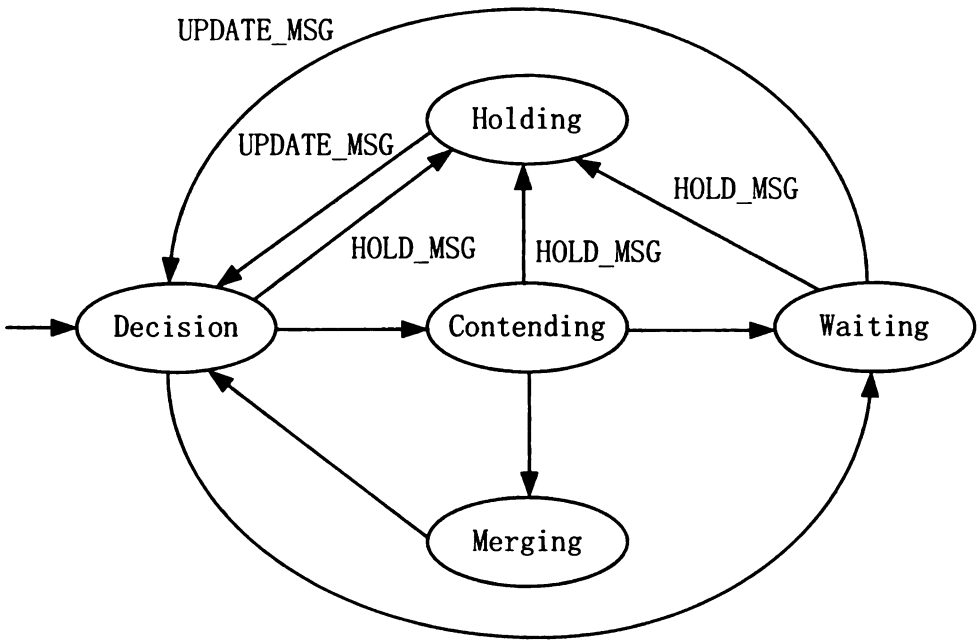


Figure 3.3: Distributed Partitioning Algorithm

enters *Waiting* state. In the *Merging* state, it first broadcasts a *HOLD_MSG* to the neighboring groups of A_x and A_y to put them into *Holding* state, and then updates its group information. After the merge is finished, it broadcasts an *UPDATE_MSG*, which contains the information of the newly merged group, to its neighboring groups. If a head node has not yet entered *Merging* state, it will enter *Holding* state when it receives a *HOLD_MSG* from neighboring groups. When a head node is in *Holding* state or *Waiting* state, it will enter *Decision* state again once it has received an *UPDATE_MSG*.

3.3.4 Probability Based Partitioning

In the previous partition algorithm, we only focused on the completely adjacent groups, that is, we ensure that each group has at least K completely adjacent groups in the partition. Thus, in the backbone network formed by activating one node from each group, each node has at least K neighbors, which maintains the network's con-

nectivit

espe ial

group. I

lem to f

Let

betwe n

Thus, if

A_i and

$0 < P_{ij}$

Wit

vious al

groups f

probabil

group h

partition

One

the back

the requ

partition

same tim

3.3.5

As all th

until its

prolong t

nectivity. However, we did not make use of the groups having arbitrary connections, especially when one group has most nodes connected with most nodes in the other group. In this subsection, we try to utilize this information in the optimization problem to further reduce the partition size.

Let A_i and A_j be two groups with size n_i and n_j. We can evaluate the connectivity between these two groups by the *connectivity probability* P_{ij}.

$$P_{ij} = \frac{m}{n_i \times n_j}, \text{where } m = \mid \{(x,y) \mid x \in A_i \cap y \in A_j \cap (x,y) \in E\} \mid$$

Thus, if A_i and A_j are completely adjacent, then $P_{ij} = 1$ because $m = n_i \times n_j$. If A_i and A_j are completely non-adjacent, then $P_{ij} = 0$ because $m = 0$. Otherwise, $0 < P_{ij} < 1$.

With slight modifications, we can utilize the connectivity probability in the previous algorithm. In Algorithm.1, we keep track of the number of completely adjacent groups for each group. Instead, we can count the number of groups with connectivity probability more than P for each group. Thus, we are able to guarantee that each group has at least K groups with connectivity probability more than P in the final partition.

One drawback of the probability based approach is that the K-connectivity of the backbone network cannot be guaranteed for sure if $P < 1$. However, by loosing the requirement on connectivity, we save more energy because of the reduced final partition size, while ensuring the K-connected backbone with high probability at the same time. We will show this in the simulation.

3.3.5 Load Balancing Energy Usage in Groups

As all the nodes in the network are equally important, running a node in active status until its energy is depleted is not an appropriate energy usage strategy. In order to prolong the lifetime of each node, the nodes in each group need to switch between

active and sleeping status periodically so that all nodes remain alive together for as long as possible.

Assume all the nodes in each group can be synchronized. In order to elect an active node, each group member broadcasts a message to the whole group stating its willingness to become active. Each node waits for a certain time delay before its announcement. The earliest announcement will suppress the others so that the corresponding node will become the active node in the group. The time delay for each node is set to be inversely proportional to its residual energy. Therefore, the node with maximum residual energy will be selected. Then, the selected active node informs other nodes of the time it will remain active, after which all the nodes need to reselect an active node again within the group.

3.4 Performance Evaluation

We evaluate our schemes under uniform deployment of sensors. The simulation is based on the energy consumption model observed in [20], that is, the ratio of energy consumed in listening, receiving, and transmitting status is 1:1.2:1.7. The initial energy level of each node is set to 500, which means the node will remain alive for 500 units of time in the listening status. According to the assumption in Section 3.2 that the energy consumption is uniform over all the sensor nodes with the mobility assisted approaches helping collect data, we simulate the energy consumption in the sensor network by an equivalent scenario. In each time slice, we randomly select 20 traffic nodes, which send and receive packets between each other. In addition, we use load balanced energy aware routing [93] in the backbone network. One slight modification we make is that we use the total residual energy of the group to denote the residual energy of the corresponding active node in the load balanced route decision.

We perform the simulation in MATLAB. The energy consumption is calculated

Processing Time

Fi

base

stat

cons

depl

the

I

ther

radi

3.4.

As C

scala

parti

In al

trans

which

unit t

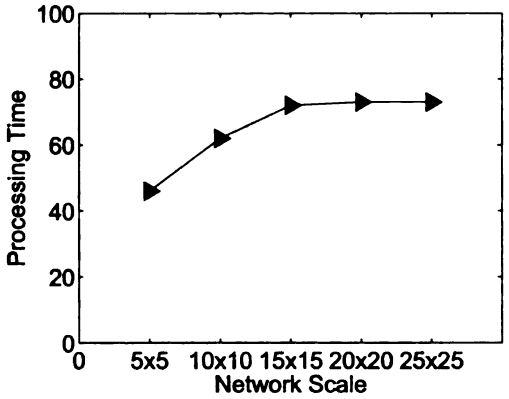


Figure 3.4: Processing Time of CPA (Node Density = 5)

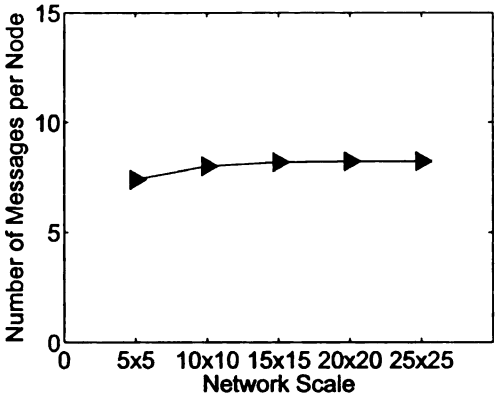


Figure 3.5: Overhead of CPA (Node Density = 5)

based on the changing status of each node and the energy consumption ratio for each status. In our evaluation of the sensor network's lifetime, we do not take the energy consumed in the partition process into consideration, because it runs only once at the deployment phase of the sensor network so that it only consumes a trivial portion of the network's total energy.

In this section, we will first analyze the scalability of our partition algorithm, and then evaluate the partition of CPA in comparison with GAF under both the ideal radio transmission model and the irregular radio transmission models.

3.4.1 Scalability of CPA

As CPA is a distributed algorithm where only local computation is involved, it is scalable with the network size. This can be seen from Fig. 3.4. We run the distributed partition algorithm on networks with the same density of sensors but different ranges. In all configurations, the density is set to 5 sensors per square unit, and the radio transmission range is set to $\sqrt{5}$. We try networks with area of 5×5, ..., to 25×25, which are shown in the x-axis of the figure. Assume that each group merge needs one unit time. We use a relative scale to measure the running time. The y value of each

point mea
time used
that the r
Fig.3.5
node dens
erage num
the averag
network si
each node
when there
groups dea

3.4.2 C

In this sub
propagatio
deploy 500
network li
our simula

In the
in differen
$\sqrt{5}$, which
to partitio
also run C
locations
for parame
the final p
this partiti

point means that the running time of the distributed algorithm is comparable to the time used for sequentially merging groups y times. We can observe from the figure that the running time tends to converge with the increase of network scale.

Fig.3.5 illustrates the overhead of CPA under different network scales while the node density remains unchanged at 5. We evaluate the overhead by counting the average number of messages transmitted per node. From the figure, we can observe that the average number of messages transmitted per node converges with the increasing network size. This is because the CPA algorithm only requires local information at each node. The overhead is reasonable, because CPA is not running frequently. Only when there is dramatic environmental change or there are a considerable number of groups dead will CPA be re-executed.

3.4.2 CPA under Ideal Radio Propagation Model

In this subsection, we evaluate the partitions of CPA and GAF under the ideal radio propagation model in a 10×10 area. We first set the node density to 5, that is, we deploy 500 sensors uniformly in the area. We compare CPA and GAF based on the network lifetime and the connectivity of backbone networks. After that, we repeat our simulation for different node densities and network scales.

In the ideal radio propagation model, the radio transmission range is the same in different directions. In our simulation, the radio transmission radius R is set to $\sqrt{5}$, which is the same with GAF. GAF uses squared cells with length of $R/\sqrt{5} = 1$ to partition the deployed area, thus all the nodes are divided into 100 groups. We also run CPA, which is based on the connectivity between nodes instead of their locations under the same experiment setting. CPA is executed with different values for parameter *mindeg*, which controls the minimum degree of the 2-induced graph of the final partition. CPA guarantees that the backbone network generated based on this partition will be *mindeg*-connected.

Partit
Approa
Numbe
Groups
Averag
Group
STD
Group

The

increase

time in

that eac

backbon

As show

is fewer

sufficient

total ene

groups. w

that each

of group

the group

result. It

under ran

We th

plied. Ea

The node

periodical

Table 3.1: Partitions of GAF and CPA

Partition Approach	CPA (min-deg=2)	CPA (min-deg=3)	CPA (min-deg=4)	GAF	CPA (min-deg=5)	CPA (min-deg=6)
Number of Groups	71	84	91	100	106	116
Average Group Size	7.0	6.0	5.5	5.0	4.7	4.3
STD on Group Size	0.92	0.77	0.79	0.73	0.67	0.63

The partition results are shown in Table 3.1. For CPA, the number of groups increases with the parameter *mindeg*, because more active nodes are needed each time in order to ensure higher connectivity of the backbone network. GAF ensures that each node is connected with at least four nodes in its four neighboring cells in the backbone network. Therefore, we can regard GAF as comparable to CPA(*mindeg*=4). As shown in the table, CPA(*mindeg*=4) partitions the nodes into 91 groups, which is fewer than GAF. This indicates that CPA can identify redundant nodes more sufficiently than GAF. As discussed in previous sections, it is preferable to have the total energy evenly distributed in the groups so as to prevent the early death of some groups, which may disrupt the connectivity of the backbone network. We assume that each node has the same initial energy when deployed, so the standard deviation of group size can be an indication of how evenly the total energy is distributed in the groups. Table 3.1 shows the standard deviation of group size for each partition result. It shows that our approach can divide energy as evenly in groups as GAF under random uniform distribution of nodes.

We then evaluate the lifetime of the network when the partition results are applied. Each group keeps only one active node each time to form a backbone network. The nodes in each group balance the energy usage by reselecting the active node periodically, which may change the topology of the backbone network. We refer to

Figure

lifetime
to be di
active n
network
fraction
backbon
the long
active n
higher n
network.
has long
same lev
network
cause the
nodes re

Altho
the topol

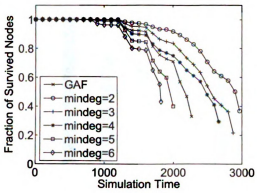


Figure 3.6: Comparison of Network Lifetime

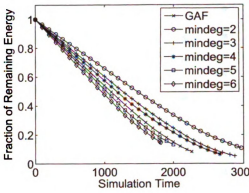


Figure 3.7: Comparison of Energy Consumption

lifetime as the time when the backbone network formed by active nodes turns out to be disconnected. If no sleeping schedule scheme is used, that is, all nodes keep active until death, the network lifetime will be less than 500. Fig. 3.6 illustrates the network lifetime where different partitions are adopted. As shown in the figure, the fraction of surviving nodes decreases with time, and the simulation stops when the backbone network becomes disconnected. The partition of CPA(*mindeg*=2) achieves the longest network lifetime (around 3000), because it keeps the fewest number of active nodes each time. The network lifetime decreases for partitions of CPA with higher *mindeg* values which can, however, ensure better connectivity of the backbone network. We can also observe from the figure that the partition of CPA(*mindeg*=4) has longer network lifetime than the partition of GAF, even though they ensure the same level of connectivity. Fig. 3.7 illustrates the energy consumption of the whole network with regard to time. The energy consumption rate is relatively constant because the traffic nodes generate traffic at a constant speed and the number of active nodes remains constant each time as well.

Although lowering node density reduces the energy consumption of the network, the topology change may affect the network's communication quality. For example,

Figure 3
on Rou

if a packe
in the or
shows the
original n
mindeg he
quite low
schedulin
delay.

We rep
range of r
is set to
changed,

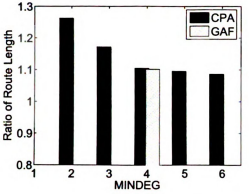


Figure 3.8: Comparison of the Impact on Route Length under GAF and CPA

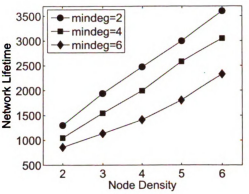


Figure 3.9: Network Lifetime under Different Node Densities

if a packet goes through a much longer path in the backbone network than it does in the original network, longer data transfer delay will be experienced. Fig. 3.8 shows the ratio of the average routing path length in the backbone network and the original network for different partitions. The ratio decreases for CPA with higher *mindeg* because its partition ensures higher connectivity. As can be seen, the ratio is quite low (below 1.3) for all the partitions listed in the figure, which means the node scheduling based on these partitions does not dramatically increase packet delivery delay.

We repeat our experiments under different node densities without changing the range of network. Table 3.2 shows the partition sizes when the parameter *mindeg* is set to different values. We can observe that when the node density remains unchanged, the partition size increases with *mindeg*, which is consistent with previous

Table 3.2: Partition Size of CPA under Different Node Densities

	mindeg=2	mindeg=3	mindeg=4	mindeg=5	mindeg=6
Density=2	66	76	88	99	110
Density=3	70	81	90	103	115
Density=4	69	81	92	104	115
Density=6	73	84	94	105	118

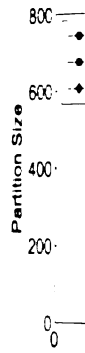


Figure 3
ferent N

experimen

approxim

gorithm c

We al

are appli

figure tha

node den

network li

unchange

nodes nee

In Fig.

run CPA v

As shown

the networ

which is sl

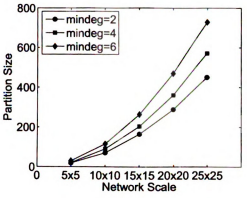


Figure 3.10: Partition Sizes under Different Network Scales (Node Density = 5)

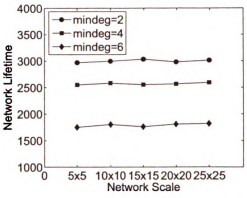


Figure 3.11: Network Lifetime under Different Network Scales (Node Density = 5)

experiment results. In addition, when *mindeg* is fixed, the partition size remains approximately the same under different node densities, which indicates that our algorithm can effectively identify redundant nodes under different node densities.

We also repeat the simulations to see the network lifetime when these partitions are applied. The simulation results are shown in Fig. 3.9. We can observe from the figure that when *mindeg* is fixed, the network lifetime increases with the increase of node density. This is because more redundant nodes can be utilized to prolong the network lifetime under higher node density. Moreover, when the node density remains unchanged, the network lifetime decreases with the increase of *mindeg*, because more nodes need to be active in each time slice to maintain better network connectivity.

In Fig. 3.10 and Fig. 3.11, we fix the node density to 5 nodes per square unit, and run CPA with mindeg equal to 2, 4, and 6 respectively on networks of different scales. As shown in Fig. 3.10, the partition size increases with the network scale. However, the network lifetime remains approximately the same regardless of the network scale, which is shown in Fig. 3.11.

3.4.3 (

In order t
environme
lower bou
the maxim
propagat
and lower

Like th
area with
in the rad
consequen
based on t
sizes under
CPA with
observe fr
communic
the radio
level of co

We p
CPA(*mu*
that is, 10
ularity. W
of average
GAF is no
boring cel
model, the
an active

3.4.3 CPA under Irregular Radio Propagation Model

In order to evaluate the performance of CPA in comparison with GAF under complex environments, we choose the DOI model [46]. This model assumes an upper and lower bound on the signal propagation range. The parameter DOI is defined as the maximum radio range variation per unit degree change in the direction of radio propagation. In our simulation of radio irregularity, we set the upper bound to $\sqrt{5}$ and lower bound to half of upper bound.

Like the previous subsection, we first perform the simulation in a 10×10 square area with the node density of 5. GAF cannot adapt to different levels of irregularity in the radio propagation model, because it is based on the sensors' locations and consequently cannot detect the irregularity level. Unlike GAF, CPA partitions sensors based on their measured connectivity, which enables it to obtain appropriate partition sizes under different levels of radio irregularity. Fig.3.12 shows the partition sizes of CPA with different *mindeg* under irregular radio with different DOI values. We can observe from the figure that the partition size increases with the DOI value. As the communication between sensors is more seriously influenced by higher irregularity in the radio propagation model, more active nodes are needed to maintain the same level of connectivity in the backbone network, leading to larger partition size.

We perform simulations to study the network lifetime under GAF and CPA(*mindeg* = 2 or 4). For GAF, we use the same partition for different DOI values, that is, 100 groups with cell length of 1, because GAF is unaware of the radio irregularity. We run the simulation multiple times for each partition, and the comparison of average lifetime is shown in Fig. 3.13. Our simulation finds that the lifetime for GAF is not stable through repeated simulations. As the connectivity between neighboring cells is no longer guaranteed by GAF under the irregular radio propagation model, there is a possibility that the backbone network formed by randomly selecting an active node from each group is disconnected. The higher the radio irregularity,

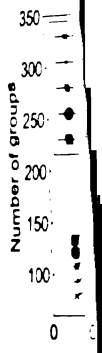


Figure 3.

the higher

figure, wh

CPA works

radio irreg

connectivi

unit. On t

side length

lower boun

groups with

the networ

As we

cent by en

under irreg

backbone n

CPA partit

under diffe

backbone n

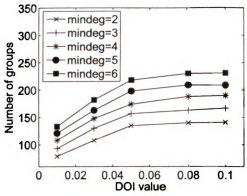


Figure 3.12: Partition Sizes under Different DOI Values

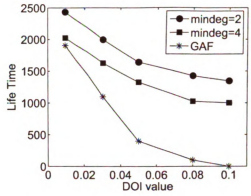
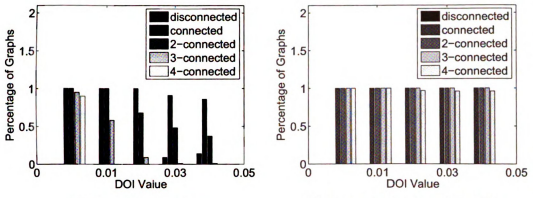


Figure 3.13: Network Lifetime under Different DOI Values

the higher the probability of a disconnected backbone network. As illustrated in the figure, when DOI is close to 0.1, the GAF partition cannot even work. In contrast, CPA works well under different conditions. The lifetime for CPA decreases with the radio irregularity level, because more active nodes are needed to maintain the same connectivity of the backbone network, and thus more energy is consumed per time unit. On the other hand, GAF does work if it divides the deployed area by cells with side length of 0.5 so that any nodes in two neighboring cells are within $\sqrt{5}/2$, the lower bound of radio transmission range in DOI model. However, this results in 400 groups with an average group size of 1.25. Apparently, this partition can only prolong the network's lifetime for a very small portion.

As we cannot guarantee the K-connectivity of the backbone network for 100 percent by ensuring the minimum degree of K in the 2-induced graph of the M-partition under irregular radio propagation models, we try to evaluate the connectivity of the backbone networks generated by CPA partitions through simulation. We select the CPA partition ($mindeg = 4$) and compare its connectivity with the GAF partition under different values of radio irregularity. For each partition, we generate different backbone networks by randomly selecting an active node from each group. We then

comp
show
comm
valu
com
we c
cally
perc
incre
0.05.
forme
the co
4-comm
CPA ar
In ac
of differ
backbon



(a) Using GAF Partition

(b) Using CPA(mindeg=4) Partition

Figure 3.14: Connectivity of Backbone Networks under Irregular Radio

compute the connectivity of each backbone network, and the statistical results are shown in Fig.3.14. The percentages of graphs that are disconnected, connected, 2-connected, 3-connected and 4-connected under the irregular radio with different DOI values are shown as bars with different colors in the figure. Note that if a graph is K-connected, it is also *(K-1)*-connected, *(K-2)*-connected and so on. From Fig.3.14(a), we can observe that the connectivity of the backbone graphs deteriorates dramatically with the increase of radio irregularity for the GAF method. For example, the percentage of 4-connected backbone graphs drops to around zero when the DOI value increases to 0.03; the percentage of 3-connected graphs approaches zero when DOI is 0.08. Furthermore, when DOI increases to 0.08, there is possibility that the backbone formed by GAF is disconnected. In contrast, as shown in Fig.3.14(b) CPA maintains the connectivity of backbone graphs much better. It keeps higher than 95 percent of 4-connected graphs even when DOI increases to 0.10, and all backbones generated by CPA are at least 3-connected under different DOI values.

In addition to the connectivity metric, we also evaluate the communication quality of different backbone networks by the ratio of average routing path length in the backbone network and the original network, because it reflects the packet delivery

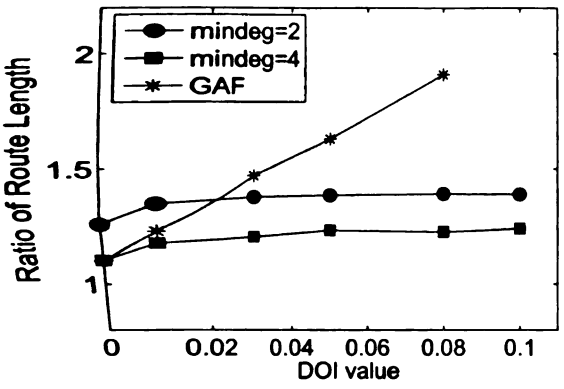


Figure 3.15: Impact on Route Length under Irregular Radio

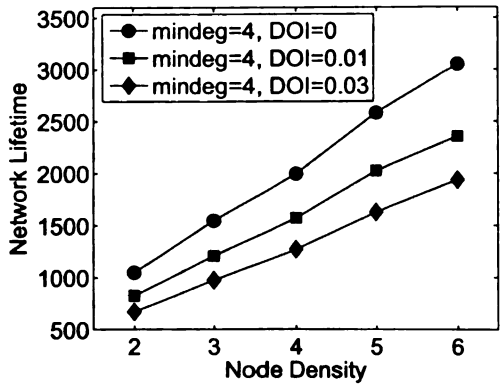


Figure 3.16: Effect of Node Density

delay. As illustrated in Fig. 3.15, the ratio of route length under the GAF method increases dramatically with the DOI value. When DOI reaches 0.1, the ratio is infinite because the backbone network is frequently disconnected. In comparison, the ratio of route length under the CPA method remains approximately constant with the change of the DOI value. The results show that CPA can form well-connected backbone networks, which preserve the communication quality under different levels of radio irregularity. Therefore, it is more adaptive to the real environment than GAF.

We change the number of nodes while keeping the same scale of the deployed area. As shown in Fig.3.16, we simulate CPA ($mindeg = 4$) in the same deployed area with 200, 300 to 600 nodes separately, that is, the corresponding densities are 2, 3, to 6 nodes per unit square. For all the cases, nodes are distributed uniformly. We evaluate the CPA partitions and the corresponding network lifetime under both ideal (DOI = 0) and irregular (DOI = 0.01, 0.03) radio propagation models. Fig.3.16 illustrates that the network lifetime under CPA partitions increases with node density in both ideal and irregular radio environments. We can also observe that under certain node density, the network lifetime decreases with the incresae of radio irregularity, because CPA needs to keep more active nodes in order to maintain the same level of network

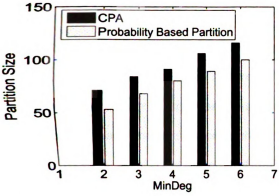


Figure 3.17: Partition Size of CPA and Probability Based Approach

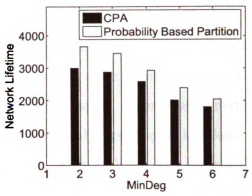


Figure 3.18: Network lifetime of CPA and Probability Based Approach

connectivity.

3.4.4 Probability Based Partition

In this subsection, we will evaluate the probability based partition algorithm discussed in Section 3.3.4. In our simulation, the deployed area is still 10×10 square with 500 nodes uniformly distributed, and the ideal radio propagation model is assumed. The threshold P is set to 0.90. Fig. 3.17 shows the partition sizes of both approaches under different configurations of *mindeg*. The probability based approach generates smaller-size partitions than CPA because it looses the requirement on the network connectivity. Fig. 3.18 illustrates the network lifetime corresponding to both approaches. The probability based approach is able to increase the network lifetime by over 10% at the cost of lowering the network connectivity a little bit. Through simulation of the probability based approach, we find that the backbone network is *mindeg*-connected with probability over 0.90 and *(mindeg-1)*-connected with probability over 0.99.

3.5 Summary

As the energy consumption of idle listening nodes is comparable to active nodes that send and receive packets in a wireless sensor network, node scheduling mechanisms can reduce energy dissipation dramatically. In this chapter, we propose to partition the nodes based on their measured connectivity instead of geographic locations. We formulate it as a constrained optimal graph partition problem, and present CPA (Connectivity Based Partition Approach) to approximate a good partition. As a distributed algorithm, CPA has fast converging speed and is scalable with network size. CPA partition outperforms GAF in two aspects. First, CPA can guarantee K-vertex connectivity of the backbone network under ideal radio propagation models, which balances the trade-off between saving energy and preserving the network's communication quality, while GAF only ensures 4-connectivity. In addition, CPA can also ensure K-vertex connectivity of the backbone network with high probability under irregular radio propagation models. Simulation results have shown that CPA increases the network lifetime to over 3 times of GAF when the radio irregularity is greater than 0.05. Therefore, CPA has better adaptivity to complex environments.

CHAPTER 4

Overlapping Channel Assignment for Wireless Mesh Networks

One major problem facing WMNs is the capacity reduction caused by the interference among multiple simultaneous transmissions [51]. When two nearby wireless links communicate on the same frequency band, they cannot transmit data simultaneously. As a result, the throughput of each link may be decreased dramatically due to the interference from the other link. Also, a router cannot transmit and receive simultaneously with a single radio. To alleviate these problems, in many WMNs, mesh routers are equipped with multiple radios, which can be configured to operate on different channels. Thus, nodes are able to transmit and receive simultaneously in a multi-radio multi-channel wireless mesh network.

To improve the throughput of WMNs, much research has been done on configuring the network interfaces of mesh routers with different non-overlapping channels to avoid interference. It is known that 802.11b/g and 802.11a provide 3 and 12 non-overlapping channels respectively. Although 802.11a has more channels, 802.11b/g is more commonly used because of its longer communication range. Due to the limited number of non-overlapping channels available by 802.11b/g, the interference cannot

be completely eliminated. However, this problem can be alleviated by considering partially overlapping channels of 802.11b/g in channel assignment.

There are 14 channels available in 802.11b/g, of which only the first 11 channels are permitted in US. According to 802.11b/g, if the channel separation is greater than 4, the two channels are non-overlapping channels. Otherwise, they are partially overlapping channels. Thus, the number of non-overlapping usable channels is at most three (channel 1, 6, and 11). Previous algorithms only consider the non-overlapping channels in the channel assignment. A simplified interference model is usually assumed, that is, if two links are within interference range of each other (twice the transmission range R), they can transmit and receive simultaneously only if they use different non-overlapping channels. As a result, the frequency spectrum has not been fully exploited in these cases.

In this chapter, we study how to further mitigate the effects of interference in 802.11b/g mesh networks by fully exploiting the spectrum resource, that is, utilizing both non-overlapping channels and partially overlapping channels, and efficient channel assignment algorithms. The contributions of this chapter are four-fold.

- We model the interference between both non-overlapping and partially overlapping channels based on our experimental results of interference measurement.
- We formulate the optimal channel assignment problem by using partially overlapping channels with the goal of maximizing the overall network throughput.
- We present a greedy channel assignment algorithm in 802.11b/g mesh networks, which fully utilizes the spectrum resource. Moreover, we propose using genetic algorithms to solve the channel assignment problem. We discuss the proper design of the genetic algorithm for channel assignment, and demonstrate that it can obtain better results than the greedy algorithm.

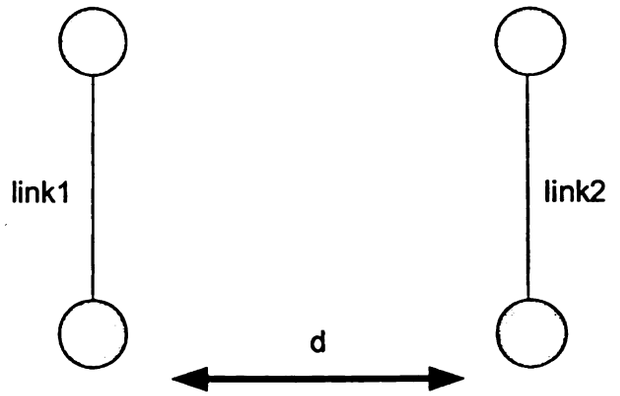


Figure 4.1: Experiment Setup

- We evaluate our channel assignment algorithms by NS2 simulations, and compare them with previous work.

The rest of the chapter is organized as follows. Section 4.1 presents the system model, including the derivation of the new weighted conflict graph model and a formulation of the channel assignment problem. In Section 4.2, we propose two channel assignment algorithms, the greedy algorithm and the genetic algorithm, and discuss on the design details. The performance evaluation is shown in Section 4.3. We conclude our work in Section 4.4.

4.1 System Model and Problem Formulation

In this section, we first experimentally study the interference between partially overlapping channels. A similar experiment in [73] studied the interference of wireless links between mobile stations and APs. In the following experiment, we study the interference between peer-to-peer wireless links. After that, we model the interference between both non-overlapping and partially overlapping channels and formulate the optimal channel assignment problem.

4.1.1 Interference Measurement

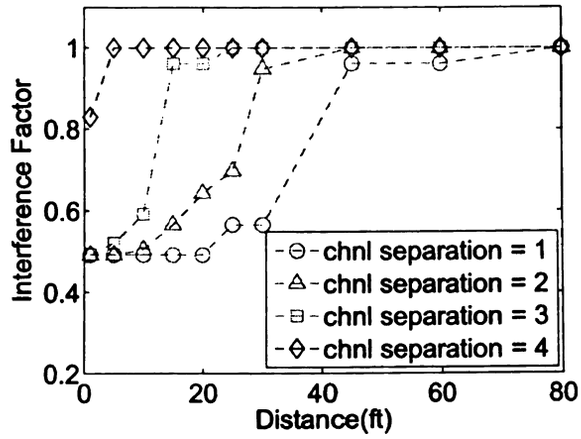
Fig.4.1 illustrates our experimental setup. We used four laptops, each equipped with a Netgear WAG511 802.11a/b/g PC Card. Linux kernel with Madwifi is used to drive the network cards. Paired laptops form a communication link, that is, the two end nodes of the link work on the same channel. We configured the two links with different channels and varied the distance between them. In this experiment, we aim to measure the level of interference between links configured with partially overlapping channels in 802.11b/g (2.4GHz). We used UDP packets in data transmission.

We used the following metric to evaluate the effect of interference. Let s_1 and s_2 be the throughput of *link*1 and *link*2 respectively when the other link is turned down. Let s_1' and s_2' be the throughput of *link*1 and *link*2 when both links are active. Then, the interference between these two links can be evaluated by *IF (the Interference Factor)* .

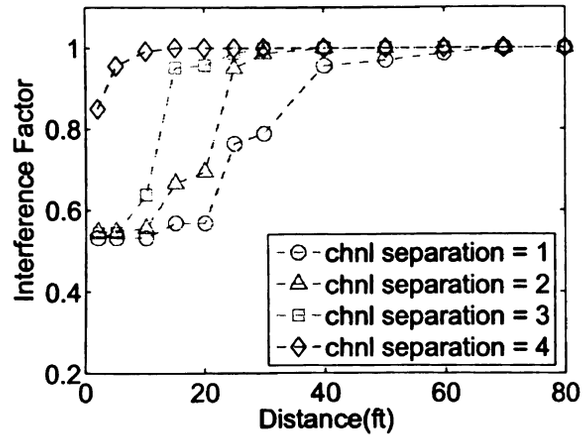
$$IF = \frac{s_1' + s_2'}{s_1 + s_2}$$

Therefore, if $IF = 1$, there is no interference between these two links, and they can transmit or receive simultaneously. When $IF < 1$, there exists interference between them. The lower IF value is, the higher the interference.

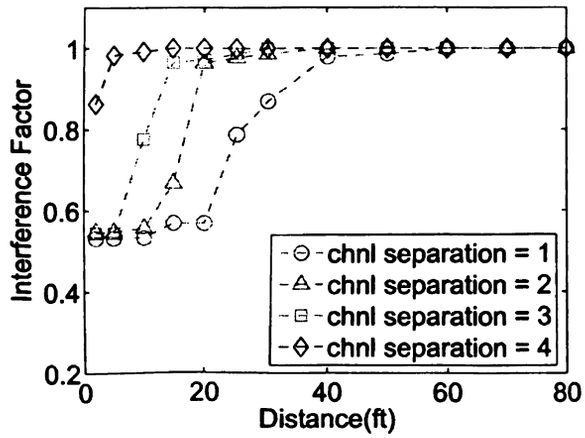
The experimental results are shown in Fig.4.2, where we can see the variance of interference factor with the physical distance and the channel separation of the two links. Each point corresponds to the average of 20 measurements, where each last 30 seconds. Fig.4.2(a), Fig.4.2(b), and Fig.4.2(c) illustrate the relationship when the bit rate of network cards are set to 2M, 5.5M, and 11M respectively. There are four possible values of channel separation for partially overlapping channels. We repeated our experiment for all possible pairs of partially overlapping channels. From the figure, we can observe that when the channel separation is fixed, interference decreases with increasing distance. When the distance is fixed, interference decreases with the increase of channel separation. Note that the results may vary slightly with



(a) BitRate = 2M



(b) BitRate = 5.5M



(c) BitRate = 11M

Figure 4.2: Interference with regard to Physical Distance and Channel Separation

different network cards. However, we observed the same trend.

4.1.2 Interference Range

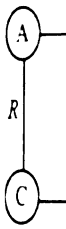
As shown in Fig.4.2, when the channel separation is fixed, the interference between two links can jump from severe interference (IF around 0.5) to almost no interference (IF around 1) with only a slight increase in distance. Thus, we can use the binary relationship to approximate the interference as follows. Let I_c be the interference range of two links with channel separation c. When the channel separation of the two links is c, they will interfere with each other if their distance is less than I_c, and otherwise not.

For example, when two links use the same channel, that is, their channel separation is 0, they will not interfere with each other as long as their distance is over $2R$, where R is the radio transmission range. When two links use non-overlapping channels, that is, their channel separation is 5, they will not interfere with each other no matter how close they are. Thus, $I_0 = 2R$ and $I_5 = 0$, which is consistent with the traditional interference model.

The interference ranges for different channel separations under different bit rates are shown in Table.4.1. The data is obtained from Fig.4.2. We use a threshold of 0.95, that is, for each channel separation, we find the minimum distance such that IF exceeds the threshold as the interference range. In our experiments, the radio transmission range R is $40ft$. The reason we use binary approximation is that the optimal channel allocation problem with non-overlapping channels is already NP hard.

Table 4.1: Interference Range

	I_0	I_1	I_2	I_3	I_4	I_5
2M	2R	1.125R	0.75R	0.375R	0.125R	0
5.5M	2R	R	0.625R	0.375R	0.125R	0
11M	2R	R	0.5R	0.375R	0.125R	0



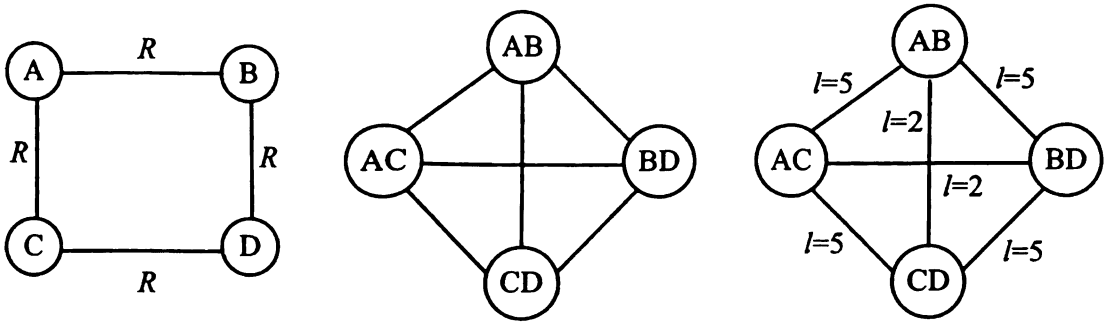
(a) Netw

By sac r:
the char

4.1.3

Conflict
$G(V, E)$
denoting
that has
overlapp
distance
of each o
distance
traditiona
overlappi

A We
conflict g
e_i-e_j are



(a) Network Topology (b) Conflict Graph (c) Weighted Conflict Graph

Figure 4.3: An Example of Weighted Conflict Graph

By sacrificing a little accuracy, we are able to design more efficient algorithms to solve the channel allocation problem with partially overlapping channels.

4.1.3 Weighted Conflict Graph

Conflict graphs have generally been used to model interference in previous work. Let $G(V, E)$ represent a wireless mesh network with V denoting mesh routers and E denoting wireless links. A conflict graph $F(S, T)$ of $G(V, E)$ is defined as a graph that has each vertex $s_i \in S$ corresponding to each link $e_i \in E$. When only non-overlapping channels are considered, the conflict graph F has an edge $s_i s_j \in T$ if the distance between the two corresponding links $e_i, e_j \in E$ is within interference range of each other (usually $2R$). The distance between two links is defined as the minimum distance between any node of one link and any node of the other link. However, the traditional conflict graph does not accurately model the interference between partially overlapping channels.

A Weighted Conflict Graph of $G(V, E)$ is denoted as $\langle F(S, T), l \rangle$, where F is the conflict graph of G, and l is a label on T. Let $s_i, s_j \in S$, whose corresponding links e_i, e_j are within $2R$ away. Then, we have $s_i s_j \in T$. Denote the distance between

these two links by $d(s_i s_j)$. The label l is defined as follows:

$$l(s_i s_j) = min\{c \mid d(s_i s_j) \geq I_c\}$$

$l(s_i s_j)$ actually indicates the minimum channel separation that links e_i and e_j must have so that they will not interfere with each other.

An example of the Weighted Conflict Graph is shown in Fig.4.3. Fig.4.3(a) shows a simple example of network topology, where R is the radio transmission range. Fig.4.3(b) illustrates the conflict graph. Assume the bit rate we are using is 2M. The weighted conflict graph is shown in Fig.4.3(c). As link $A-B$ and $A-C$ share a common node, they must use non-overlapping channels in order to avoid interference. In order words, their channel separation should be no less than 5, $l(V_{A-B}V_{A-C}) = 5$. On the other hand, the distance between links $A-B$ and $C-D$ is R. According to Table.4.1, as long as their channel separation is greater than or equal to 2, they will not interfere with each other. Thus, we have $l(V_{A-B}V_{C-D}) = 2$.

4.1.4 Minimum Interference Channel Assignment

Since we are focusing here on channel assignment algorithms, we assume that the network topology has been determined through careful planning or by some topology control algorithms beforehand such as [90] and [50]. We abstract the mesh network topology as a graph $G(V, E)$, where each vertex represents a mesh router, and each edge represents a wireless link. Each pair of mesh routers of a link has a separate interface devoted to constructing the link. Similar to [50], we assume the wireless mesh network has dynamic unicast traffic, that is, each connection demand has random sources, destinations and arrival times. This is because there will be substantial random and unpredictable traffic within the mesh network caused by peer-to-peer and newly emerging applications in addition to the traffic from and towards the Internet.

To maximize the overall network throughput, it becomes the following problem: *"Given enough channel resources, assign the vertices of the weighted conflict graph with channels (or colors) while satisfying that the channel separation of adjacent vertices is no less than the weight on the edge between them, such that the span between the minimum and maximum channel used is minimized."* This problem can be modeled as T-Coloring problem [43], which is NP-hard. In reality, the channel resource that we can use is usually limited, so our goal becomes minimizing the interference in the network with limited channel resource. The channel assignment problem can be formulated as follows.

Let $\langle F(S,T), l\rangle$ be the weighted conflict graph of $G(V,E)$ and C be the set of channels. We define a label A on S, $A(s_i) \in C$ is the channel on which link $s_i \in S$ is working. We also call A as a channel assignment scheme for the wireless mesh network.

Let $I(s_i, s_j, A(s_i), A(s_j))$ be the interference indicator between links $s_i, s_j \in S$, that is, it indicates whether these two links will interfere with each other under channel assignment A. For the simplicity of presentation, we define the following two objective functions:

$$H_1(\langle F, l\rangle, A) = \sum_i \sum_{j \neq i} I(s_i, s_j, A(s_i), A(s_j))/2 \tag{4.1}$$

$$H_2(\langle F, l\rangle, A) = \max_i \sum_{j \neq i} I(s_i, s_j, A(s_i), A(s_j)) \tag{4.2}$$

H_1 defines the total interference within the network, that is, the total number of link pairs that interfere with each other, while H_2 denotes the maximum link interference. Therefore, we are trying to find the channel assignment A that minimizes H_1 or minimizes H_2. These problems are NP hard, because the graph coloring problem is an NP complete problem.

In this chapter, we consider two general traffic patterns: i) There are a fixed number of flows in the network, and each flow wants to achieve as higher throughput

as it can. ii) Each flow has the requirement of a fixed data rate. We want the network to support as many flows as it can. By minimizing H_1, we reduce the contentions cause by simultaneous transmissions in the network, which improves the overall network throughput. Therefore, it is more appropriate to be used for the first traffic pattern. By minimizing H_2, we maximize the capacity of the bottleneck link, which increases the opportunity for each flow to find paths with enough bandwidth. Therefore, it is more appropriate to be used for the second traffic pattern.

4.2 Channel Assignment Algorithms

In this section, we focus on channel assignment algorithms that maximize the overall network throughput, that is, to minimize the objective functions H_1 and H_2. In the following, we will solve for approximate solutions with objective H_1. The proposed algorithms can be easily modified for objective H_2. The solutions based on both objectives will be evaluated.

We propose two algorithms, the greedy algorithm and the genetic algorithm, for channel assignment using partially overlapping channels. By presenting the greedy algorithm, we want to show how the network interference can be further minimized by using partially overlapping channels, because previous work such as [90] [50], also uses greedy strategies for channel allocation with non-overlapping channels. Genetic algorithms have not been used to solve the channel assignment problems before, but they have the potential to find better solutions than the greedy algorithm.

4.2.1 Greedy Algorithm

Generally speaking, our greedy algorithm is a series of decisions, each of which assigns a channel to a link, until all links have been assigned channels. Each decision is usually composed of two steps – *select* and *assign*. In the *select* step, a link that has not been

assigned a channel is chosen according to metric α, and in the *assign* step, a proper channel is assigned to the selected link according to metric β. In each step, the link and its channel selection are determined by maximizing (or minimizing) their corresponding metrics α and β.

We define the metric α of a link as the expected level of interference between this link and all the other links in the network. As during the greedy channel assignment process, some links may not have been assigned channels yet, we use the expected value to evaluate the interference when selecting a link. Given the weighted conflict graph $\langle F(S,T), l \rangle$, the expected interference of link $s \in S$, denoted by $\alpha(s)$, is computed as follows:

$$\alpha(s) = \sum_{s' \in S_1} \bar{I}_1(s, s') + \sum_{s' \in S_2} \bar{I}_2(s, s')$$

where S_1 is the set of links that have already been assigned channels, and S_2 is the set of links not assigned channels yet. $\bar{I}_1(s, s')$ (or $\bar{I}_2(s, s')$) denotes the expected interference between s and s', which has been assigned (or not assigned) a channel. They are calculated in the following ways:

$$\bar{I}_1(s, s') = \frac{1}{\mid C \mid} \sum_{i \in C} I(s, s', i, A(s'))$$

$$\bar{I}_2(s, s') = \frac{1}{\mid C \mid^2} \sum_{i,j \in C} I(s, s', i, j)$$

Therefore, in each step, we select the link s that has the minimum expected interference $\alpha(s)$.

In the *assign* step, we define the metric $\beta(c)$ for each candidate channel c that can be assigned to the selected link. $\beta(c)$ indicates the interference between the selected

Algorithm 2 Greedy-Algorithm($\langle F(S,T), l\rangle$, C)

1: $A(s) = null$ for all $s \in S$
2: $S_1 = \{\}$, $S_2 = S$
3: **while** $S_2 \neq \{\}$ **do**
4: Calculate $\alpha(s)$ for each link $s \in S_2$
5: Select s^* such that $\alpha(s^*) = \min_{s \in S_2} \alpha(s)$
6: For link s^*, calculate $\beta(c)$ for each channel $c \in C$
7: Select c^* such that $\beta(c^*) = \min_{c \in C} \beta(c)$
8: $A(s^*) = c^*$
9: $S_1 = S_1 \cup \{s^*\}$, $S_2 = S_2 - \{s^*\}$
10: **end while**

link and those links already assigned channels.

$$\beta(c) = \sum_{s' \in S_1} I(s, s', c, A(s'))$$

We select the channel c that has the minimum $\beta(c)$, thus minimizing the interference added to the network when we assign channels to the selected link.

As described in Algorithm 2, given the weighted conflict graph $\langle F(S,T), l\rangle$ and the channel set C, the greedy algorithm obtains a channel assignment scheme A. This algorithm is able to find a solution very fast because it never changes a link's channel once it is assigned. Next, we will present a genetic algorithm, which has the potential to obtain near-optimal results.

4.2.2 Genetic Algorithm

Genetic algorithms are adaptive heuristic search algorithms premised on the ideas of natural selection and genetic evolution. Its basic concept is to simulate the process of evolution in the natural system, during which fitter individuals are more likely to survive so that they appear in the next generation.

The genetic algorithm is applicable to the channel assignment problem because of the following reasons:

- The channel assignment problem has an inherent *local optimization* property. In other words, a good channel assignment scheme for a subnetwork that causes less interference locally is more likely to be contained in a good channel assignment scheme for the entire network that causes less interference globally. (For example, we study a population of 5000 random channel assignment schemes for a 100 node network topology. We pick up two different channel assignment schemes randomly and compare their sub-schemes for a subnetwork. Experiments show that if one scheme is better than the other, its sub-scheme for the subnetwork is also better than the other sub-scheme with a probability of 76.1%.) This property fits well into the genetic algorithm [84]. In the genetic algorithm for channel assignment, we regard a channel assignment scheme for a subnetwork as local DNAs, and regard a channel assignment scheme for the entire network as an individual. Thus, we improve local DNAs to form better individuals in the genetic algorithm.

- In the genetic algorithm, individuals from the current generation are selected to breed the next generation, which is expected to have fitter individuals with high probabilities. This *fitness preservation* property [15] exists in the channel assignment problem, because we find that we can obtain better channel assignment schemes in each evolution by the genetic algorithm (as shown in Fig.4.6) We observe that better channel assignment schemes are more likely to be created by combining the good "parts" (channel assignment for subnetworks) of the channel assignment schemes in the current generation.

1. Problem Mapping

In order to solve the channel assignment problem by genetic algorithms, the first step is to establish a mapping between them. In our channel assignment problem, we denote a channel assignment for a single link as a ***DNA***, and a channel assignment

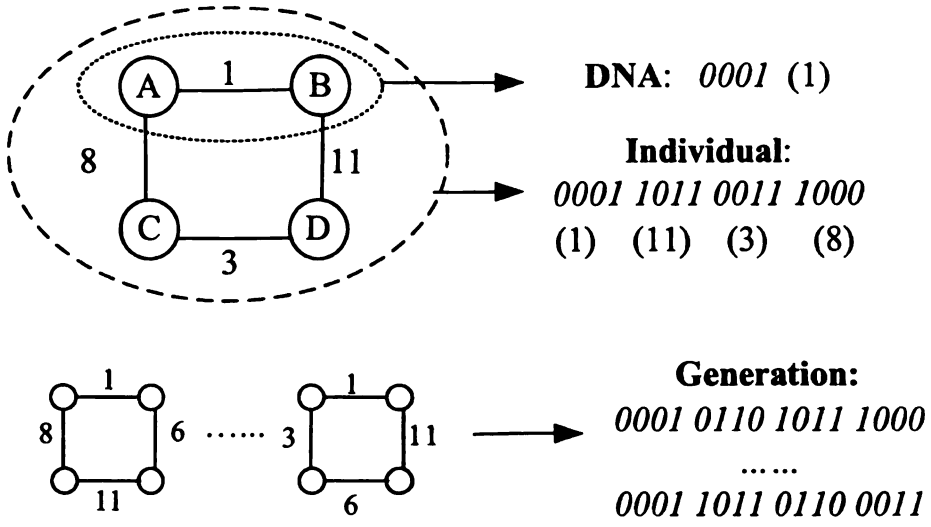


Figure 4.4: Problem Mapping

scheme for all links of the entire network as an ***individual***, which is as well a solution to the channel assignment problem. In the genetic algorithm, a generation is composed of a set of individuals. Thus, we define a ***generation*** as a set of channel assignment schemes (a set of solutions) in our problem. An example of the problem mapping is illustrated in Fig.4.4. The input of the problem is a network topology with 4 nodes and 4 links, where each node has two interfaces. The numbers beside the links represent channels. The channel assignment of each link is considered as a DNA. The channel assignment scheme shown in the top figure is considered as an individual, and the set of channel assignment schemes shown in the bottom figure constitutes a generation.

Each channel assignment scheme is encoded as a binary string in the genetic algorithm. The transformation is in the following way.

1. Sort the links in linear order;

2. Convert the channel assigned to each link into a binary string with a fixed length (a DNA);

3. Concatenate all the binary strings one by one into a single binary string (an individual) according to the order of the links.

For example, Fig.4.4 shows the binary representation of each individual. The links are sorted in the order of AB, BD, CD, AC. The channel assigned to each link is encoded into a string with a fixed length of 4.

The order of the links has a great impact on the performance of the genetic algorithm. Intuitively, we prefer the links that are within interference range of each other to be close together in the sorted list. As a result, when we combine "parts" of individuals to create offsprings, the local optimization property of the parent individuals (good channel assignment for subnetworks) can be preserved in the next generation. There are several strategies that can be used for link ordering: i) random order, ii) breadth-first search (BFS), iii) depth-first search (DFS) iv) inductive order, which has been used in graph coloring [49]. In our genetic algorithm for channel assignment, we find that BFS and DFS give better performance (shown in Section 4.3). This is because both the breadth first order and depth first order try to keep neighboring links close together in the encoding, while the random order and inductive order do not.

A genetic algorithm is equipped with a ***fitness function***, which is used to evaluate the fitness of an individual. The fitter an individual is, the better the solution. Given a binary string representing a channel assignment scheme for a network, we define its fitness function f as the negative of the total interference within the network if the channel assignment scheme is applied (the less interference, the higher fitness value and therefore better solution).

2. Algorithm Design

Our genetic algorithm begins with an initial generation, which is composed of N randomly generated channel assignment schemes. Each next generation is generated by

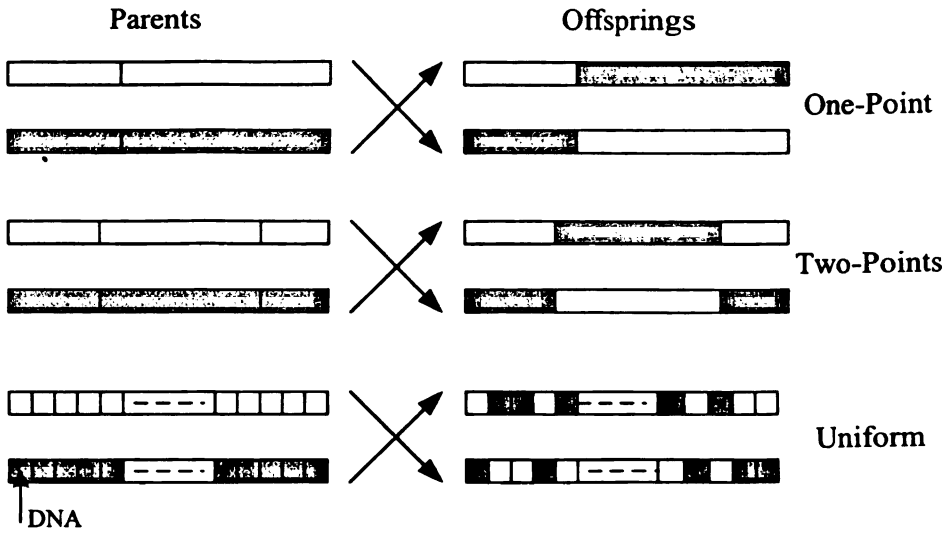


Figure 4.5: Crossover Strategies

a series of *selection-reproduction* steps from the current generation. In the ***selection*** step, two individuals in the current generation are picked up. Two constraints must be satisfied: i) Fitter individuals are more likely to be selected. ii) The other individuals also have a proper possibility to be selected, which guarantees an extent of diversity. The maintenance of diversity plays an important role, because it can avoid early convergence on local optima (the local optima has been observed in the channel assignment problem [98]). In the ***reproduction*** step, the two selected individuals breed two new individuals through *crossover* and *mutation*. *Crossover* is a strategy to combine parts of parent individuals to produce new individuals and *mutation* further changes part of a new individual with a small probability, which is called mutation rate, to enhance diversity.

Our genetic algorithm for solving the channel assignment problem is depicted in Algorithm 3, where N denotes the initial population size (the number of individuals in the first generation) and M is the number of generations to evolve. The algorithm has polynomial running time, because there are $MN/2$ steps, each of which requires polynomial time.

Algorithm 3 Genetic-Algorithm($\langle F, l \rangle$, C)

```
1: Randomly generate N channel assignment schemes, P
2: Calculate f value for each scheme in P
3: while M loops are not completed do
4:   while N/2 loops are not completed do
5:     Select p1, p2 from P using selection strategy
6:     Create o1, o2 from p1, p2 using reproduction strategy
7:     Calculate f(o1), f(o2)
8:     Find two channel assignment schemes b1, b2 that have the lowest f values in
       P
9:     Replace b1, b2 with o1, o2
10:  end while
11: end while
12: Return the fittest channel assignment scheme in P
```

- *Selection* strategy: As the channel assignment problem has many local optima, we adopt the stochastic selection, which is good at maintaining population diversity. This strategy exploits two well-studied selection methods, the roulette wheel selection and the tournament selection. In roulette wheel selection, each individual is selected with a probability proportional to its fitness function. Individuals with higher fitness are more likely to be selected, while individuals with low fitness still has a chance to be selected. In tournament selection, n individuals are selected at random and the fittest one of them is chosen. In stochastic selection, we first selects 2 individuals independently by using roulette wheel selection, and then choose the better one of them according to tournament selection.

- *Reproduction* strategy: It includes *crossover* and *mutation*. When two individuals are selected to breed offsprings, both are split into several parts by cutting at some random points. Offsprings are produced by combining different parts from the two parent individuals. With respect to the number of cutting points, well-known crossover methods include one-point crossover, two-points crossover and uniform crossover. In uniform crossover, each DNA is swapped between parent

individuals with a fixed probability, typically 0.5. Fig.4.5 illustrates different crossover methods. In the figure, two parents are drawn in different colors, and offsprings contain different parts from parents. Uniform crossover is more predominantly used due to its superior ability to produce a wide range of genes [39] [48]. However, we find that the one-point crossover works the best with respect to our channel assignment problem (shown in Section 4.3). This is due to the local optimization property of the channel assignment problem. Uniform crossover disperses the DNAs from parent individuals into the new individual, and thus does not preserve the good channel assignment schemes for local subnetworks in the new individual. To determine the proper mutation rate, it has been recommended that mutation rate be differed by an order of magnitude of 0.001 on binary-encoded continuous-valued optimization problems, and the mutation rate be in the range $[0.005, 0.01]$ [94] [44].

- In addition, we have also used some other important techniques to further improve the performance, including *elitism, sharing and niching.* By elitism, we select individuals with a bias towards the better ones as parents to generate offsprings. By fitness sharing and niching, we maintain good population diversity by degrading an individual's fitness based on the number of similar individuals in the population. For a complete discussion on genetic algorithms, please see [71].

4.3 Performance Evaluation

In this section, we evaluate our algorithms in NS2. The Hyacinth extension [6] was used to support multiple channels and multiple interfaces per node in the simulator. We made some further extensions to support partially overlapping channels in addition to non-overlapping channels. We use the 802.11 MAC protocol. In all the

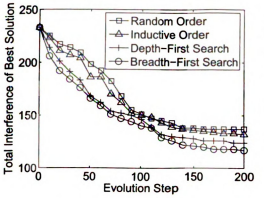


Figure 4.6: Edge Ordering

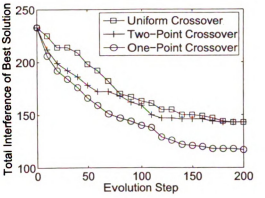


Figure 4.7: Crossover Strategy

simulations, the bit rate of each interface is set to $11Mbps$. The radio transmission range is set to $250m$, and the interference ranges for different channel separations are set in the simulator based on the experimental results in Table 1.

4.3.1 Impact of Parameters on Genetic Algorithm

Before presenting the simulation results, we first analyze several parameters that may influence the performance of the genetic channel assignment algorithm.

In Section 4.2.2, we introduced several edge ordering strategies. In order to analyze which strategy is better, we run the genetic algorithm with different edge orders on regular graphs with different sizes. Fig.4.6 shows the result for a 7×7 grid graph with 49 nodes. The x-axis denotes the number of generations in the evolution process, and the y-axis shows the best solution of each generation (measured by the total network interference). Each point corresponds to the average of 20 runs. We can observe that with the evolution of generations, the genetic algorithm with BFS or DFS edge order converges faster (getting better results in each evolution) than random order and inductive order, while BFS is slightly better than DFS. We find the similar phenomenon in networks of other sizes. Fig.4.7 illustrates the impact of different crossover strategies on the genetic channel assignment algorithm. We run the genetic

algorithm with one-point, two-point and uniform crossover strategies on the same regular graph. From the figure, we can observe that the algorithm with one-point crossover has the fastest converging speed.

For the simulation in the rest of the chapter, we use BFS edge ordering and one-point crossover strategy for the genetic algorithm. For the other parameters, we set the population size to 5000, the mutation rate to 0.005, and the number of generations to 300.

4.3.2 Simulation Methodology

In this subsection, we evaluate our channel assignment algorithms in Section 4.2. We compare the following algorithms: i) Greedy algorithm with non-overlapping channels [50]. ii) Tabu-based search algorithm with non-overlapping channels [98]. We choose the algorithm for comparative evaluation, because it is the most recent centralized static channel assignment algorithm under the assumption that the network topology has been priori determined. The algorithm starts from a random channel assignment, generates a sequence of channel assignments, and pick up the best assignment as the final solution. From the current channel assignment in the sequence, it generates a set of channel assignments, where each is obtained by varying the channel assigned to a random vertex, and then pick the best solution from the set to be the next channel assignment in the sequence. To achieve fast convergence, it avoids reassigning the same color to a vertex more than once by maintaining a tabu list, which records the history of varying channels assigned to vertices. iii) Genetic algorithm with non-overlapping channels. iv) Greedy algorithm with all channels. v) Tabu-based search algorithm with all channels (we modified the Tabu-based search algorithm to support the assignment of overlapping channels in the same way as the greedy algorithm). vi) Genetic algorithm with all channels. We run each algorithm under two different optimization goals: i) Minimizing the total interference; ii) Minimizing the maximum

link interference.

We compare these algorithms based on the following metrics.

- The value of the "objective function" (the total interference or maximum link interference) is a simple metric to evaluate an algorithm's capability of solving for optimal solutions.

- Average link bandwidth: Assume each node has equal possibility to send out data, and thus we can measure the bandwidth of each wireless link in the network. If the network is without any interference, the bandwidth of each link should be its bit rate. However, with limited channel resources, the network interference cannot be completely eliminated, which causes the link bandwidth to decrease dramatically.

- Overall network capacity: We assume each connection demand has random source and random destination. We evaluate the channel assignment schemes calculated by different algorithms based on two traffic patterns: i) There are a fixed number of flows in the network, and each flow wants to achieve as higher throughput as it can. In this case, we calculate the total throughput of all the flows. ii) Each flow has the requirement of a fixed data rate. In this case, we calculate the maximum number of flows the network can satisfy. A flow can be satisfied if we can find a path with the required bandwidth for it. In all the simulations, we use UDP for each flow, and set the packet size to 1024 bytes.

For all the figures in the rest of chapter, we use "N-O" to denote "non-overlapping" channels.

4.3.3 Minimizing Total Interference

We first study the scenario of regular topologies, because wireless mesh networks are usually deployed after careful planning, and the regular topology can average the

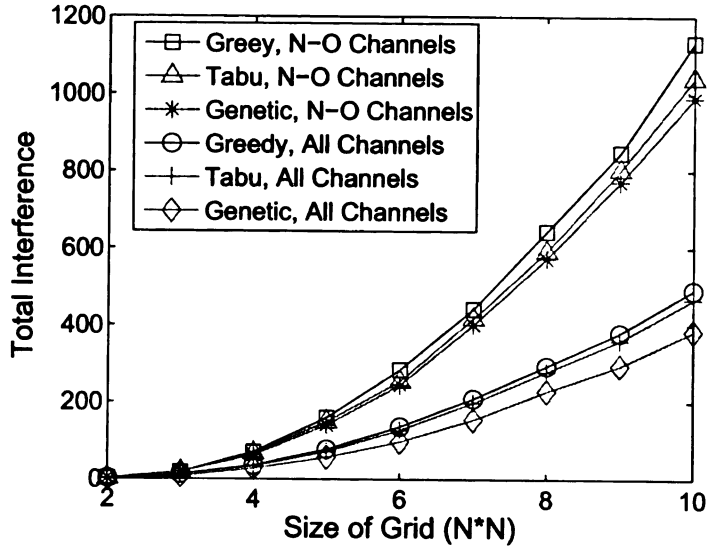
performance over the whole network. We use the topology of $N \times N$ squared grids with side length of 200 meters, that is, each vertex is deployed with a mesh router, and each edge denotes a wireless link. Each inside mesh router is equipped with 4 network interfaces that communicate with its 4 neighbors independently, while each boundary router is equipped with 3 or 2 interfaces depending on its number of neighbors.

For the first traffic pattern, we set 20 flows in the network, where each flow has random source and random destination. Each flow tries to transmit data at the highest rate that it can. In this case, the optimization goal of minimizing the total network interference is more appropriate. By minimizing the number of interfering pairs of links, we reduce the contention in the network, and therefore increase the throughput of all the flows.

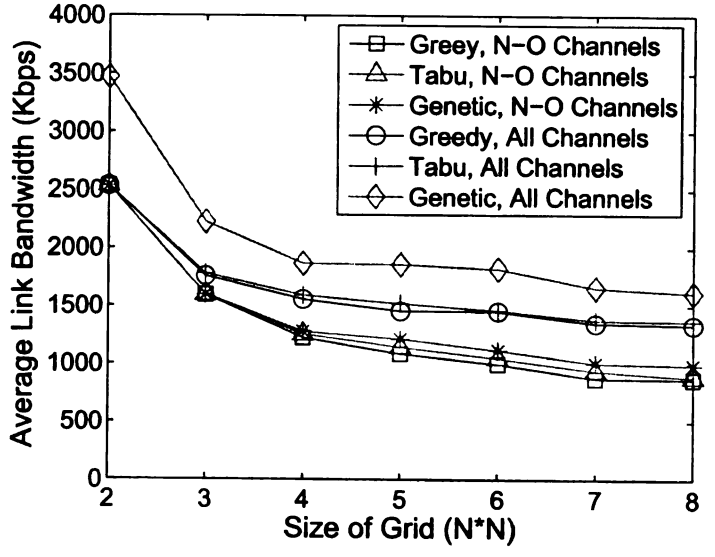
1. Total Interference

Fig.4.8(a) illustrates the total interference within networks of different sizes for different algorithms. We can observe that when only non-overlapping channels are used, the genetic algorithm finds better solutions (with smaller total interference) than the other two, while the Tabu-based algorithm is slightly better than the greedy algorithm. We find the same trend when all channels are used. The only difference is that the improvement of the genetic algorithm is more dramatic when all channels are used, compared to when only non-overlapping channels are used. This is because we have more choices of channels (11 channels) to assign to each link when all channels are used instead of only 3 non-overlapping channels. In this case, the greedy or Tabu-based search algorithm becomes less efficient. The figure demonstrates that the genetic algorithm has the potential to find more optimal results than the other two algorithms under the same condition, especially when all channels are used.

When the same algorithm strategy is used, the total interference can be decreased by using partially overlapping channels in addition to non-overlapping channels. Take



(a) Total Interference



(b) Average Link Bandwidth

Figure 4.8: Channel Assignment under Regular Topology

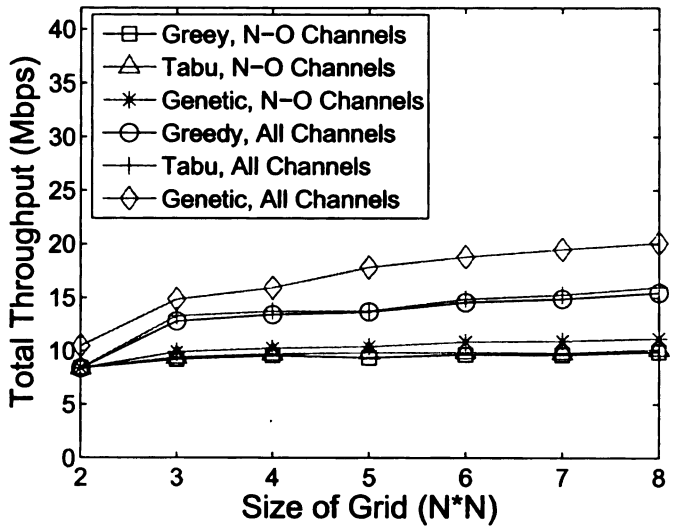
the genetic algorithm for example. With the increase of network size, the total interference for the genetic algorithm with all channels is less than 50 percent of the genetic algorithm with only non-overlapping channels. Among all the algorithms, the genetic algorithm with all channels finds the best solution under different network sizes. It decreases the total interference by over 70 percent, compared to the worst algorithm (the greedy algorithm with non-overlapping channels).

2. Average Link Bandwidth

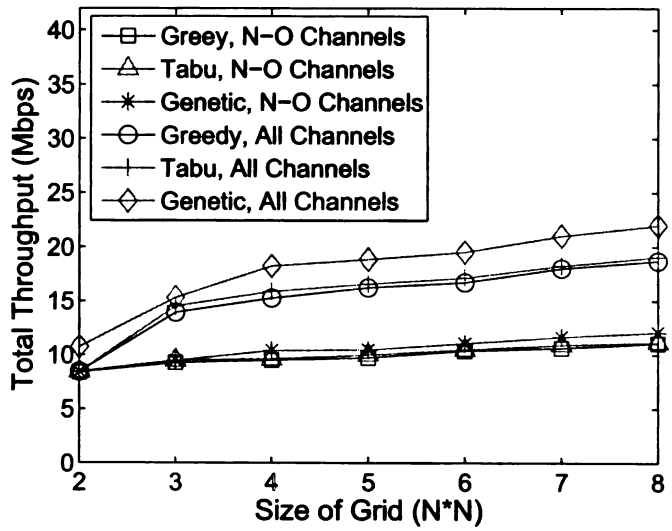
Fig.4.8(b) compares the average link bandwidth of the network for different algorithms. The Tabu-based search algorithm has only slightly higher average link bandwidth than the greedy algorithm under the same condition. For both algorithms, the average link bandwidth can be improved by around 30 percent by using partially overlapping channels when the network becomes large, which implies that the network can carry more traffic. This is because the network interference can be decreased by using partially overlapping channels. When all channels are used, the genetic algorithm further increase the average link bandwidth by approximately 20 percent, because the genetic algorithm finds better channel assignment schemes with lower total network interference than the other two algorithms.

3. Overall Network Throughput

We use two routing algorithms to determine the route of each flow: i) Shortest Path (SP) routing: This routing algorithm only depends on the network topology. Thus, the routing algorithm generates the same route for the same flow under different channel assignment schemes. This enables us to compare the different channel assignment schemes under exactly the same condition. ii) $WCETT$ routing [85]: The routing algorithm considers the channel diversity on the path. Therefore, the same flow may have different routes under different channel assignment schemes in the same topol-

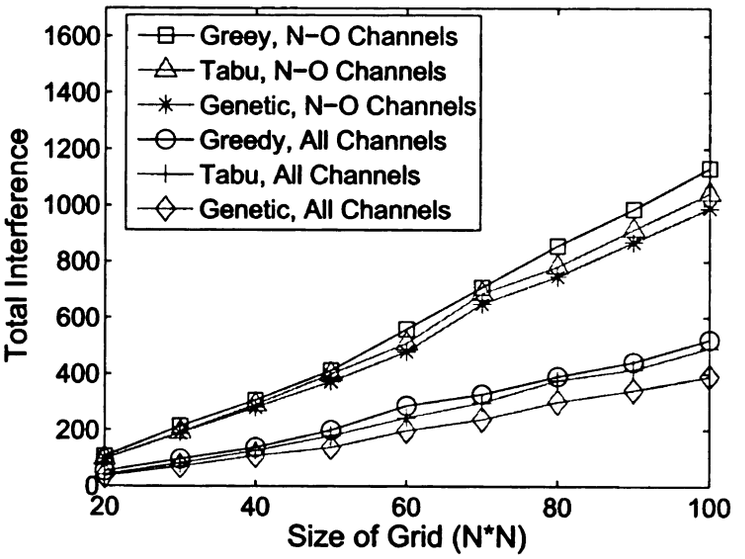


(a) Throughput (SP routing)

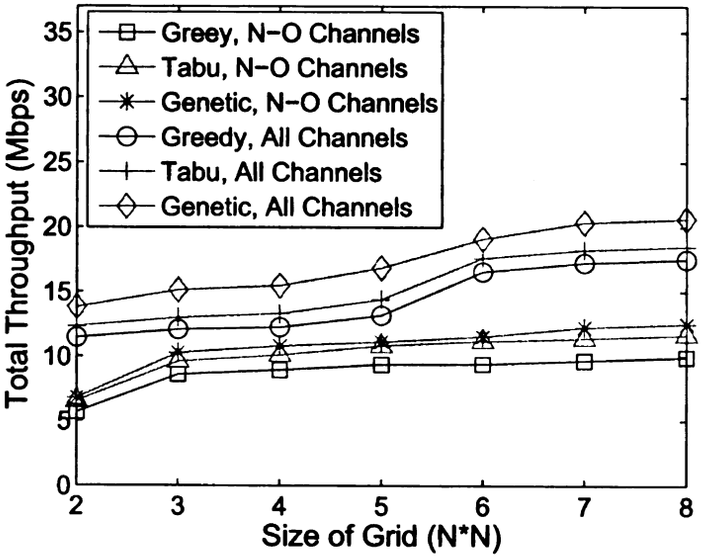


(b) Throughput (WCETT routing)

Figure 4.9: Network Throughput under Regular Topology



(a) Total Interference



(b) Throughput (WCETT routing)

Figure 4.10: Random Topology (DEG=4)

ogy. This routing algorithm aims at finding a high-throughput path for each flow, and therefore is a better match for the target of satisfying the first traffic pattern.

Fig.4.9(a) illustrates the network throughput under different channel assignment algorithms when SP routing is used. For the greedy algorithm and the tabu-based search algorithm, if we fully exploit the channel resources, we can achieve an improvement of network throughput by approximately 40 percent. In addition, the genetic algorithm achieves 20 percent higher network throughput than the other two algorithms when all channels are used. Fig.4.9(b) illustrates the network throughput when WCETT routing is used. We can observe the same trend for different channel assignment algorithms. Note that we can achieve higher network throughput by using WCETT routing compared to SP routing under the same condition.

In addition to the regular topology, we also evaluate our algorithms based on random topologies. We use the topology generation tool *GT-ITM* [3] to generate random topologies. We choose the Locality Model to generate flat random graphs with uniformly distributed nodes, and obtain graphs with desired scales and average degrees. We generate random graphs of different scales but with the average degree of 4. The simulation results are shown in Fig.4.10. In the figures, each point corresponds to the average of running each algorithm on 20 random graphs. We get the similar trends with the regular topology cases. The only difference is that the performance improvement is not that dramatic. The network throughput can be improved by approximately 25 percent if we fully exploit the spectrum resource. Moreover, the throughput can be further improved by 15 percent using the genetic algorithm instead of the other algorithms when all channels are used.

4.3.4 Minimizing Maximum Link Interference

For the second traffic pattern, we set the rate of each flow to $400Kbps$. If we can find a route with the required bandwidth for a flow, then the flow is satisfied and

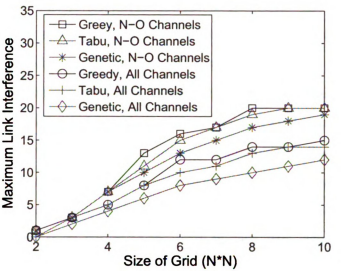


Figure 4.11: Maximum Link Interference

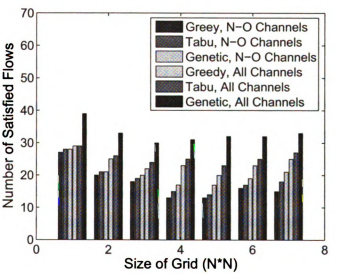


Figure 4.12: Network Capacity

we establish the flow in the network; otherwise, it is blocked. The network capacity can be defined as the maximum number of concurrent flows that the network can satisfy. In this traffic pattern, the optimization goal of minimizing the maximum link interference is more appropriate, because it can maximize the bandwidth of bottleneck links in the network, so that the network connectivity is not easily broken when more and more flows have been established in the network.

In Section 4.2, we introduced both the greedy algorithm and the genetic algorithm with the goal of minimizing the total interference within a wireless mesh network. The two algorithms can be easily modified in order to minimize the maximum link interference (in other words, minimizing the interference of the bottleneck link). For the greedy algorithm, we select and assign a channel to a link that minimizes the current maximum link interference within the network in each step. For the genetic algorithm, the fitness function is set to be the negative of the maximum link interference. In the similar way, we can also modify the Tabu-based search algorithm to optimize for this objective.

1. Maximum Link Interference

We run different algorithms with only non-overlapping or all channels on regular network topologies of different sizes. Fig.4.11 shows the maximum link interference within the network when the channel assignment schemes computed by different algorithms are applied. We can observe that better solutions can be achieved using partially overlapping channels. In addition, the genetic algorithm obtains better results than the other two algorithms, while the Tabu-based search algorithm is slightly better than the greedy algorithm.

2. Net

For eac
algorit
the po
in Fig.
of part
portin
the be

4.4

In this
can he
terfere
utilizat
lem. wh
assignm
channe
good re
polynor
algorith
40 perc
the net
of chan
work th
solution

2. Network Capacity

For each flow, we use the routing algorithm in [50] to find the route. The goal of this algorithm is to avoid using bottleneck links in the network, so that the network has the potential to support more subsequent flows. The network capacity is illustrated in Fig.4.12 for different channel assignment algorithms. We can observe that the use of partially overlapping channels can dramatically increase the network capacity (supporting more concurrent flows), and the genetic algorithm with all channels achieves the best performance.

4.4 Summary

In this chapter, we have demonstrated how the use of partially overlapping channels can help improve the network throughput. We first experimentally studied the interference between partially overlapping channels, and showed the potential of the utilization of these channels. We formulated the optimal channel assignment problem, which aims at maximizing the network performance. We proposed two channel assignment algorithms, which utilize both non-overlapping and partially overlapping channels in the channel assignment. The greedy algorithm is fast but may not get good results, while the genetic algorithm has the potential to get better results with polynomial running time. Our simulations have demonstrated that when the same algorithm is used, the overall network throughput can be improved by approximately 40 percent by using all channels than using non-overlapping channels only, because the network interference can be better mitigated. In addition, when the same set of channels are used, the genetic algorithm achieves around 20 percent higher network throughput than the previous algorithms, because it is able to solve for better solutions with less interference.

CHAPTER 5

Channel Allocation for Hybrid Wireless Mesh Networks

There are currently two approaches of channel allocation, that is, static approach and dynamic approach. In static channel allocation, each interface of every mesh router is assigned a channel permanently. In dynamic channel allocation, an interface is allowed to switch from one channel to another channel frequently. Both strategies have their advantages and disadvantages.

Static strategies do not require interfaces to switch channels, and thus have lower overhead. However, they depend on the stable and predictable traffic patterns in the network. For example, [91] [11] require that the exact traffic profile is known ahead, while [90] [50] assume known statistical traffic patterns. Dynamic strategies, such as [113] [96], require frequent channel switching, and thus have higher overhead than static strategies. However, as the channel allocation can be changed with the changing traffic, dynamic strategies are more appropriate when the network traffic changes frequently and is unpredictable.

In the real environment, the overall traffic profile is usually complex. It not only contains some predictable traffic, e.g., a large amount of traffic from end-users to the

Internet through gateways, but also contains a considerable amount of unpredictable peer-to-peer traffic between end-users due to the emerging new applications within the community. Due to the inflexibility of pure static channel allocation and the high overhead of pure dynamic channel allocation, we propose a hybrid architecture, which combines the advantages of both approaches. In this architecture, one interface from each router uses the dynamic channel allocation strategy, while the other interfaces use the static channel allocation strategy. The links working on static channels provide high throughput paths from end-users to the gateway while the links working on dynamic channels enhance the network connectivity and the network's adaptivity to the changing traffic. Therefore, this hybrid architecture can achieve better adaptivity compared to the pure static architecture without much increase of overhead compared to the pure dynamic architecture.

In this chapter, we study several important issues in the hybrid wireless mesh network.

- The system architecture. As each mesh node contains both static and dynamic interfaces, we discuss on how to coordinate the channel assignment between both types of interfaces, so that the channel resources could be utilized efficiently.

- The channel allocation for dynamic interfaces. MMAC [96] is one of the most efficient dynamic channel allocation protocols. However, the channel assignment in MMAC is optimized only for network throughput. We propose an Adaptive Dynamic Channel Allocation protocol (ADCA), which considers both throughput and delay in the channel assignment. Compared with MMAC, ADCA is able to reduce the packet delay without degrading the network throughput.

- Routing decision in the network. In the hybrid architecture, we have static links and dynamic links, both of which can be used to transmit data. We propose an Interference and Congestion Aware Routing protocol (ICAR) with the goal of

improving the overall network throughput.

The rest of the chapter is organized as follows. In Section 5.1, we propose the hybrid wireless mesh network architecture. We analyze the trade-off between throughput and delay in dynamic channel allocation algorithms in Section 5.2. In Section 5.3 and 5.4, we present our dynamic channel allocation protocol and the routing algorithm for hybrid wireless mesh networks. We evaluate our protocols in Section 5.5, and summarize our work in Section 5.6.

5.1 Hybrid Wireless Mesh Network Architecture

In this section, we propose to use the hybrid architecture to achieve both adaptivity to changing traffic and low channel switching overhead. Let $G(V, E)$ be the network topology, where V is the set of mesh routers and E represents pairs of mesh routers that are within radio communication range. Assume each mesh router has multiple interfaces. In the hybrid architecture, we let one interface of each mesh router be able to switch channel frequently, and let the other interfaces work on fixed channels. In the rest of this chapter, we call the former interface as *dynamic interface*, and the latter as *static interface.*

Fig.5.1 illustrates a hybrid multi-channel multi-radio wireless mesh network. Most mesh nodes including the gateway have 3 interfaces, and a few boundary nodes (c, g, i, f) have 2 interfaces. For each mesh node, one interface works as dynamic interface, and the others work as static interfaces.

The channel allocation of static interfaces aims at maximizing the throughput from end-users to gateways, which usually constitutes a major portion of the traffic in the network. Some heuristic algorithms, such as [90], could be used. In the algorithm, a load balanced tree is constructed for each gateway. The goal of the tree construction is to allocate bandwidth fairly to each user with regard to the user-gateway throughput.

After

The li

conges

these l

figure.

Dy

are wi

channe

dynam

each d

node c

dynam

is divi

data in

(or co

data in

channel

should

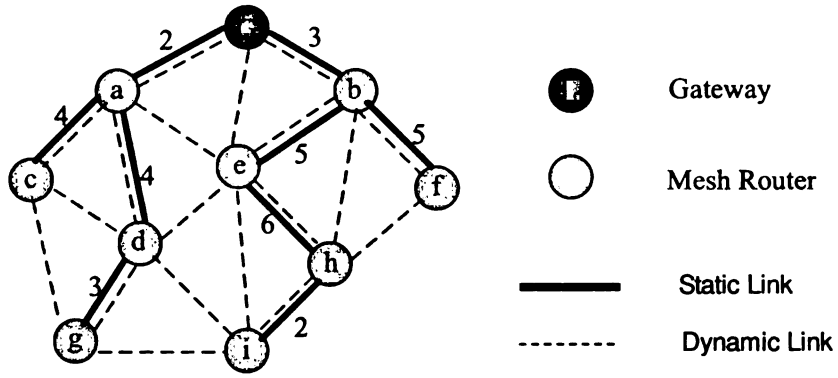


Figure 5.1: The Hybrid WMN Architecture

After the topology has been constructed, each link can then be assigned channels. The links closer to the gateways are given higher priority to be allocated with less congested channels. In Fig.5.1, the tree topology is shown in bold lines, and we call these links as *static links*. The channels assigned to static links are also shown in the figure.

Dynamic interfaces work in an on-demand fashion. Two dynamic interfaces that are within radio transmission range of each other are able to negotiate a common channel and communicate when they have data to transmit. We call these links as *dynamic links*. Fig.5.1 illustrates all possible dynamic links in dotted lines. Note that each dotted line only implies that the pair of nodes are able to communicate, but each node can only communicate with one neighbor at a time. We can use $TDMA$-style dynamic multi-channel MAC protocols such as MMAC [96] in this case. The time is divided into fixed-length intervals, each of which consists of control interval and data interval. In the control interval, all nodes communicate on a default channel (or control channel) to negotiate the channels to use in the data interval. In the data interval, nodes transmit and receive data on the negotiated channels (or data channel). As we are considering a hybrid architecture, the dynamic channel allocation should be aware of the interference from the static interfaces, which requires some

additional considerations in the protocol design.

- All dynamic interfaces need to negotiate on a default channel in each control interval. As the control interval is very important for dynamic channel allocation, we should eliminate the interference from static interfaces in this phase. Thus, we exclude this default channel in the channel allocation of the static interfaces. Note that this will not harm the channel usage efficiency of the whole system, because dynamic interfaces can still use this default channel in the data interval.

- The data channel is determined by choosing the "least congested channel" in the neighborhood of the communicating pair. As dynamic interfaces negotiate on the same default channel in each control interval, each interface can be aware of the channel usage of their neighboring dynamic interfaces by listening on the default channel for their communication demands. However, they cannot know the channel usage of nearby static interfaces in the same way. To solve this "hidden terminal problem" in the multi-channel environment, we can take advantage of the static nature of wireless mesh networks. As wireless mesh networks usually have static topology, each mesh node can maintain a stable list of other mesh nodes within its interference range. We can let each mesh node measure the channel usage of its static interfaces periodically, and send this information to all the other mesh nodes within its interference range. Thus, the dynamic interface of each mesh node is able to know the channel usage of static interfaces within its interference range.

There are several advantages of using this hybrid architecture.

- If pure static channel allocation strategies are used, the connectivity of the original network topology may be degraded, which can lead to suboptimal routing paths. By using one dynamic interface in each mesh node, the connectivity of

the network topology can be well-maintained, because each dynamic interface is able to communicate with any other interface within radio transmission range.

- Pure static channel allocation strategies cannot adapt to the changing network traffic. For example, in the tree topology, if the users of the left subtree happen to generate a much larger amount of traffic than the users in the right subtree, then the links in the left subtree get more congested. The use of dynamic links is able to direct traffic around the congested areas and therefore achieve better load balance in the network.

In the following sections, we discuss the achievement of adaptive dynamic channel allocation and the routing decision in the hybrid network separately.

5.2 Trade-off between Throughput and Delay

Consider a linear topology of 5 nodes, where each node has only one interface, as shown in Fig.5.2. Suppose we use MMAC [96] with 2 channels and the bit rate is set to $1Mbps$. In Fig.5.2, simultaneous transmissions among different links are made possible. As a result, the throughput from S to D is nearly doubled compared to the case of using single-channel MAC protocol 802.11, which is shown in Fig.5.3(a).

However, the throughput is improved at the cost of increasing packet delay. In MMAC, for example, every two nodes try to switch to a same channel to transmit data in each time interval, and thus each packet can be transmitted at most one hop away in one interval. Fig.5.3(b) illustrates the packet delay of MMAC and 802.11 when we vary the data rate of the flow from S to D. We observe that when the traffic rate is under $170Kbps$, 802.11 has much lower delay than MMAC. This is because the network traffic is not saturated, so packets can be transmitted across multiple hops very quickly. With the increasing traffic rate, the delay of 802.11 increases sharply because the network becomes over-saturated, which causes long waiting time

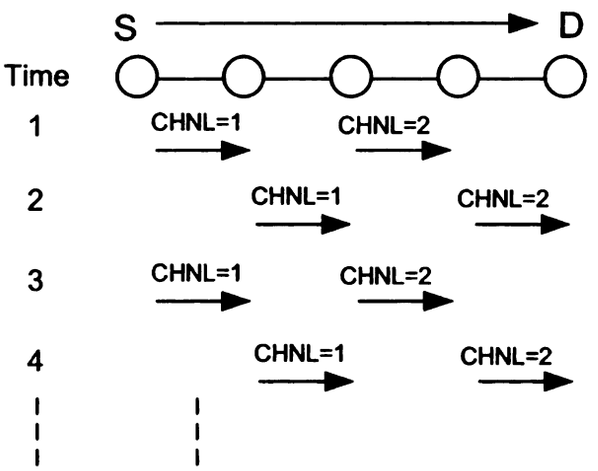


Figure 5.2: Linear Topology of Wireless Mesh Network

of packets in queues. In comparison, MMAC becomes better because the network is still below saturation with usage of multiple channels.

Fig.5.3 shows that although MMAC improves the network capacity, it is not making enough efforts to decrease packet delay when the network traffic is below saturation. In other words, the channel allocation scheme of MMAC, that is, every two nodes choosing a least congested channel to transmit data in each time interval, is only optimized for maximizing network throughput. However, when the traffic load is below saturation, a channel allocation scheme that is optimized for both throughput and delay can not only satisfy the traffic demands, but also reduce the packet delay. Therefore, it motivates us to design an adaptive dynamic channel allocation protocol, which can achieve lower packet delay while satisfying the imposing traffic.

The packet delay of MMAC is related to the length of time interval. By decreasing the length of time interval, we can reduce the packet delay at the cost of reducing the maximum network capacity. However, it is impractical to adaptively change the interval length in MMAC under changing traffic due to several reasons. 1) It is hard to measure the traffic load of the whole network in real time. 2) The network

Fi

traffic

of the

difficul

adaptiv

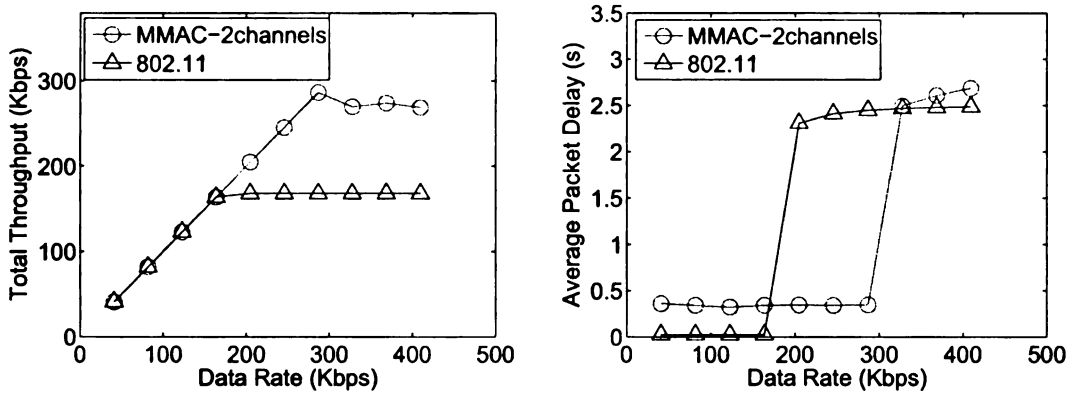
5.3

When

delay.

•

•



(a) Network Throughput (b) Packet Delay

Figure 5.3: Tradeoff between Throughput and Delay in Multi-channel MAC

traffic distribution may not be uniform. It is not feasible to make different parts of the network use different time interval lengths, which makes the synchronization difficult. In the rest of this chapter, we will propose an alternative way to achieve adaptivity in dynamic channel allocation protocols.

5.3 Adaptive Dynamic Channel Allocation

When the traffic load is below saturation, MMAC may cause unnecessary packet delay, which can be illustrated by the examples in Fig.5.4.

- In Fig.5.4(a), assume A has some data to send to C. According to MMAC, in the first time interval t_1, the packets are transmitted from A to B, and then in the second interval t_2, the packets are transmitted from B to C. Although the packets can be transmitted from A to C through B in one interval, MMAC actually requires two intervals. This is because B has to wait till the second interval to negotiate a common channel with C in order to continue transmitting the data.

- In Fig.5.4(b), assume A has some data to both B and C, we can see that MMAC

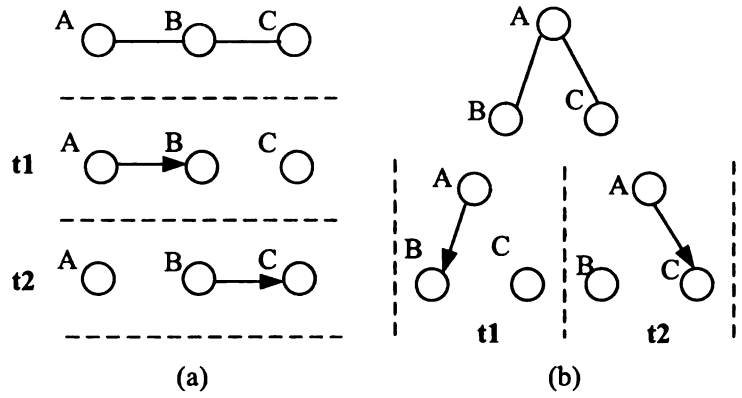


Figure 5.4: Unnecessary Delay of MMAC when Traffic is below Saturation

still needs two intervals to complete the transmission, while it can actually be done in one interval. As shown in the figure, in the first interval, A can only communicate with one of the nodes, B in this case, so it has to wait till the second interval to communicate with C.

The reason why MMAC causes unnecessary delay in the above cases is that only pairs of nodes negotiate common channels in each interval in MMAC. If we enable more than two nodes to negotiate a common channel, the transmission delay can be dramatically reduced. For example, in both cases of Fig.5.4, if A, B, and C can negotiate a common channel together, then all the transmissions can be completed in one time interval. Next, we present the design of Adaptive Dynamic Channel Allocation protocol (ADCA).

ADCA uses the similar framework with MMAC. It divides time into fixed length intervals. Each interval is further split into control interval and data interval. Let T be the interval length, and T_c, T_d be the length of control interval and data interval respectively (Obviously, we have $T = T_c + T_d$). In the control interval, all nodes switch to the same default channel and negotiate channels. In the data interval, the nodes working on the same channel transmit and receive data among each other. In MMAC, T is set to $100ms$, and T_c is set to $20ms$, which is long enough for nodes

to nego

parame

control

In A

one qu

the cor

with M

neighbor

the neigh

only cons

this strat

considerati

length and

pairs of no

in Fig.5.5(a

Differe

For a pair

initiated th

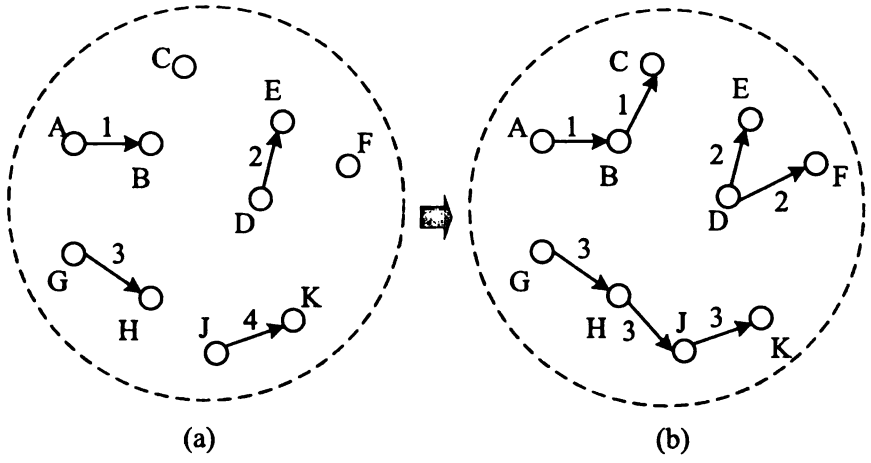


Figure 5.5: Adaptive Dynamic Channel Allocation

to negotiate channels when network traffic is saturated. Our protocol uses the same parameter settings (T and T_c), but is different in the channel allocation scheme during control interval.

In ADCA, each dynamic interface maintains multiple queues in the link layer with one queue for each neighbor. The data to be sent to each neighbor are buffered in the corresponding queue. The first step of channel negotiation in ADCA is similar with MMAC. For each dynamic interface, if it has data to transmit, it selects a neighbor that it wants to communicate and try to negotiate a common channel with the neighbor. There are many criteria for selecting neighbors. If throughput is the only consideration, we may select the neighbor with the longest queue. However, this strategy may cause starvation. Therefore, we augment it with some fairness considerations, that is, we evaluate a neighbor's priority by considering both its queue length and how long the queue has not been served. As a result, during this step, pairs of nodes have negotiated common channels with each other such as the example in Fig.5.5(a).

Different from MMAC, ADCA enables further channel negotiation among nodes. For a pair of nodes that have already negotiated a channel, denote the node that initiated the channel negotiation as *sending node*, and the other node as *receiving*

Algorithm 4 Channel Negotiation

Pending Node:

1: Broadcast $PNODE\_REQ$ message to notify its neighbors that it is a pending node.
2: **if** receiving $SWITCH\_CHNL$ **then**
3: Switch to channel c indicated in the message.
4: **end if**

Sending Node:

1: **if** its queue length for the receiving node $< QT$ **then**
2: Broadcast $SNODE\_REQ$ message to notify its neighbors that it is a sending node and the traffic load is below saturation.
3: **end if**
4: **if** receiving $SWITCH\_CHNL$ **then**
5: **if** its receiving node (r) is not negotiating with any other sending nodes **then**
6: Switch to channel c indicated in the message.
7: Notify r to switch to channel c.
8: **end if**
9: **end if**

Receiving node:

1: **if** the queue length of its sending node $< QT$ **then**
2: **if** receiving $PNODE\_REQ$ **then**
3: Send $SWITCH\_CHNL$ message to the pending node including its own channel c.
4: **end if**
5: **if** receiving $SNODE\_REQ$ **then**
6: Send $SWITCH\_CHNL$ message to the sending node including its own channel c.
7: **end if**
8: **end if**

node. The node that has not succeeded in negotiating a channel with any other node is denoted as *pending node.* The channel negotiation process is described in Algorithm 4. In the algorithm, a queue length threshold QT is used on each node to decide whether to perform further channel negotiation. If the queue length is over the threshold, the traffic load may become saturated, and it will not bring any benefit with further channel negotiation. We will discuss how to determine QT later.

The e
traffic is
ing to N
Fig.5.5
to C th
C does
same c
right t
furthe
has s
chann
case,
C
than
ADC
Wher
ADC

5.4

The pr
RTT, E
single fl
network
flows to
and the
congestio

The example in Fig.5.5 illustrates how our protocol works. Assume the network traffic is below saturation. In Fig.5.5(a), pairs of nodes negotiate channels according to MMAC. Then further channel negotiations are performed as illustrated in Fig.5.5(b). There are three cases illustrated in the figure. 1) A has some data to send to C through B. A and B got the right to transmit data on a common channel, while C does not. In this case, B can further negotiate with C so that C works on the same channel with A and B. 2) D has some data to both E and F. D and E got the right to transmit data on a common channel, while F does not. In this case, D can further negotiate with F so that F works on the same channel with D and E. 3) G has some data to J through H. G and H got the right to transmit data on a common channel, while J got the right to transmit data to K on a common channel. In this case, H can negotiate with J so that G, H, J, and K work on the same channel.

Compared with MMAC, ADCA can negotiate common channels among more than two nodes in each interval when network traffic is not saturated. As a result, ADCA has the potential to reduce packet delay while satisfying the imposing traffic. When the traffic is near saturated, ADCA will behave similar with MMAC. Therefore, ADCA is adaptive to the network traffic.

5.4 Interference and Congestion Aware Routing

The previous routing metrics proposed in wireless mesh networks include hop count, RTT, ETX, ETT, WCETT. These metrics aim at finding a good quality path for a single flow. In this chapter, our goal is to maximize the total throughput in the hybrid network with both static links and dynamic links. We want the routes of different flows to be selected efficiently such that the channel usages are balanced at each node and thereby avoiding congestion in the network. We propose an interference and congestion aware routing metric in the following.

A

in b

the

the

ban

link

the

the s

wirel

To

link l

contai

bandw

formul

The

network

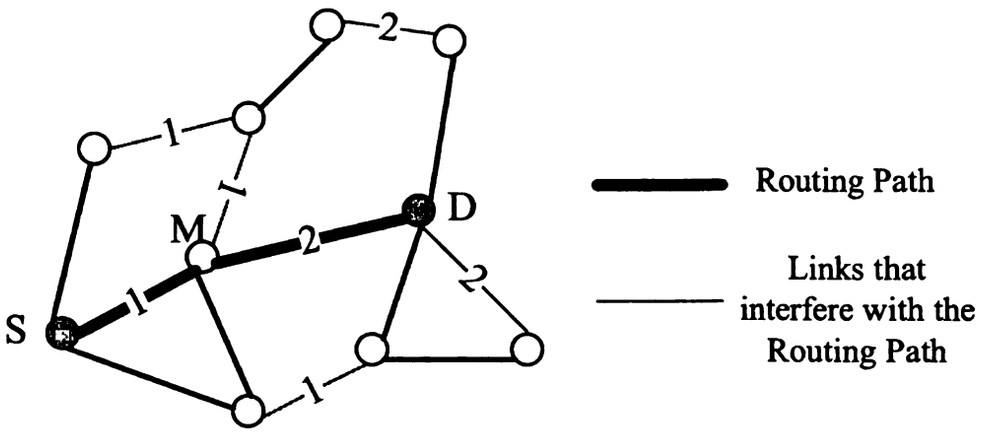


Figure 5.6: Effect of Inter-flow Interference

As shown in Fig.5.6, assume the routing path of a flow from S to D is illustrated in bold lines, and the rate of the flow is r. All the wireless links that interfere with the routing path are plotted in grey lines. Now, let us consider the bandwidth that the transmission of the flow will consume in the network. The flow will consume bandwidth of r on link SM. However, there are three other links that interfere with link SM. As interfering links cannot be active at the same time, we may think that the flow is also consuming the same bandwidth in the other interfering links. It is the similar case with link MD. Therefore, although the flow is routed through two wireless links, it is actually consuming bandwidth of multiple links in the network.

To describe it formally, let P be the routing path of a flow with rate r. For each link $l \in P$, let $IE(l)$ be the set of links that interfere with l (assume $IE(l)$ also contains l). Let $ETX(l)$ be the expected transmission count of link l. The total bandwidth that is consumed by routing the flow can be calculated by the following formula.

$$B_{total} = \sum_{l \in P} \mid IE(l) \mid * ETX(l) * r$$

The B_{total} metric reflects the resource consumed by routing the flow through the network. In order to support as many flows in the network as possible, the routing

of each

better

congest

We

gested

theref

mean

able

thus

assu

addi

$w1$

r al

The

$\sum_{e\in}$

flow.

T

state

Th

each s

queue

can be

average

Unli

channel

of each flow cannot be too selfish. This metric can help balance the channel usage better than the previous metrics. In order to make the routing better at avoiding congested links, we make some further modifications on B_{total} in the following.

We categorize each link into three states: Congested, Median, and Low. "Congested" means the channel on which the link is working is highly congested, and therefore it is not desirable to route additional traffic through this link. "Median" means the channel used by the link has a considerable amount of load, but is still able to route additional traffic. "Low" means the channel has very light load, and thus the link is preferred to be used to route additional traffic. Therefore, we can assign different weights $w1$, $w2$, $w3$ to three different states, which reflects the cost of adding additional traffic to the link in different states. Obviously, they should satisfy $w1 > w2 > w3$. By inducing the weights into B_{total}, the cost of routing a flow of rate r along a path P can be calculated as follows.

$$Cost_{total} = \sum_{l \in P} [ETX(l) * r * \sum_{e \in IE(l)} w(e)]$$

Therefore, we can define a cost metric of each link as $C(l) = ETX(l) * r * \sum_{e \in IE(l)} w(e)$ and then use source routing to find a minimum cost path for each flow, $P^* = argmin \sum_{l \in P} C(l)$.

There remains two issues in our routing algorithm, the determination of links states and interfering links.

The link states can be inferred from the queue length. In the hybrid architecture, each static interface has one queue while each dynamic interface maintains multiple queues, one for each neighbor. The state of each link, whether static or dynamic, can be inferred from the corresponding queues of the two end interfaces. The longer average queue length, the more likely is the link congested.

Unlike the static link whose channel is fixed, a dynamic link may work on different channels at different intervals. Thus, if there is a dynamic link in a pair of links,

we cannot deterministically say whether they will interfere with each or not. An alternative way is to estimate their probability of interference.

For each dynamic link, we maintain statistics of its channel usage history, in the form of $\{p_i | i = 1, 2, ..., C\}$, where p_i is the fraction of time that the link worked on channel i. Note that this representation is also applicable to static links. If the static link works on channel c, then $p_c = 1$ and $p_i = 0 (i \neq c)$ in its representation. Given two links, l_1 and l_2, whose channel usage is represented by $\{p_i^{l_1}\}$ and $\{p_i^{l_2}\}$, we can predict their probability of interference using the following formula.

$$P_{if}(l_1, l_2) = \sum_{i \in \{1,2,...,C\}} p_i^{l_1} * p_i^{l_2}$$

Therefore, the cost formula can be modified as:

$$Cost_{total} = \sum_{l \in P} [ETX(l) * r * \sum_{e \in IE(l)} (w(e) * P_{if}(e, l))]$$

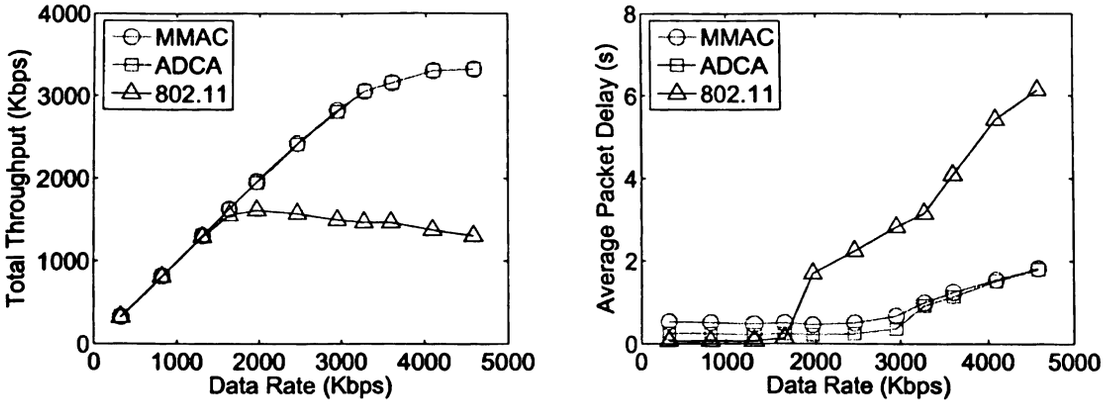
The channel usage array is easy to maintain in each node, because dynamic links change channels in a synchronous fashion, that is, interval by interval. This information can be updated to routing agents periodically.

5.5 Performance Evaluation

We performed simulations in NS-2.31. The Hyacinth extension [6] was used to support multiple channels and multiple interfaces per node in the simulator. We made further extensions to support dynamic channel switching on each interface. In all the simulations, the bit rate of each interface is set to $11 Mbps$. The radio communication range is 250 meters.

5.5.1 Evaluation of Adaptive Dynamic Channel Allocation

In this subsection, we evaluate the performance of ADCA by comparing it with MMAC.



(a) Network Throughput (b) Packet Delay

Figure 5.7: Performance of ADCA under 50 Nodes and 3 Channels

1. Improvement of Packet Delay

We first perform the simulation on a random topology of 50 nodes in a $1200m \times 1200m$ area. Each node has at least 2 neighbors and at most 8 neighbors within radio communication range. The average degree of the nodes is 5.4. To compare with MMAC, each node has only one interface, which can switch channels dynamically. We generate 25 UDP flows with the same data rate in the network, each with random source and random destination. The packet size of each flow is set to 512 bytes. We then vary the data rate of each flow, and observe the total throughput and the average packet delay over all the flows.

We analyze three protocols: 1) 802.11, a single-channel MAC protocol; 2) MMAC, a multi-channel MAC protocol; 3) ADCA, an adaptive multi-channel MAC protocol. For both MMAC and ADCA, we use 3 orthogonal channels. The queue length is set to 100 packets, and the queue threshold QT is set to 30 packets for ADCA. By using the same parameter in [96], we set the time interval to $100ms$, and the control interval to $20ms$. These three protocols are compared with regard to throughput and delay in Fig.5.7.

In Fig.5.7(a), with the increasing data rate, the network is gradually becoming

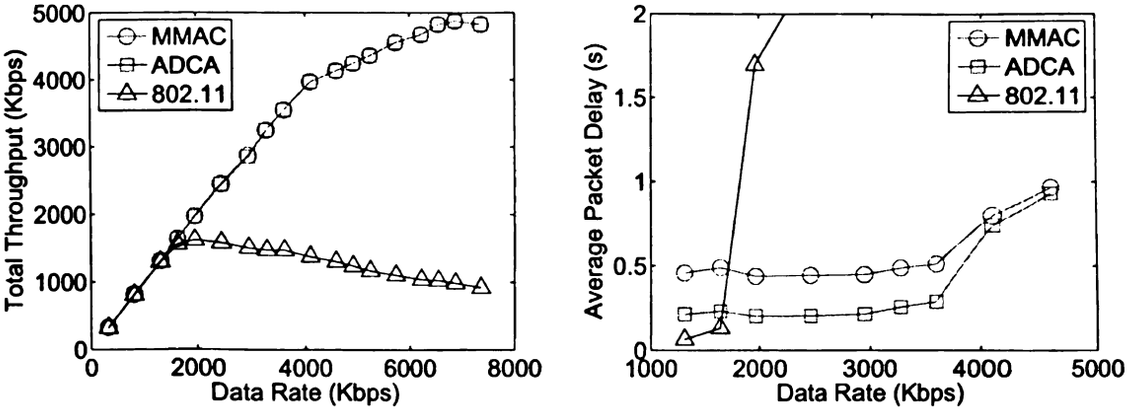
saturated. We can observe that ADCA achieves the same throughput with MMAC. The maximum throughput of ADCA and MMAC is over twice of 802.11 because of the utilization of multiple orthogonal channels.

Fig.5.7(b) illustrates the average packet delay with varying data rate. We can observe that when the traffic load is below saturation ($1600Kbps$) of single-channel MAC, 802.11 has the lowest delay. However, with the increasing traffic load, 802.11 suffers a sharp increase in delay because packets are experiencing long waiting time in queues. On the other hand, as multi-channel MAC achieves higher capacity, the delay is much lower than 802.11 when the traffic load is over $1600Kbps$. Interestingly, when the traffic load is below $1600Kbps$, 802.11 has much lower delay than MMAC.

In comparison with MMAC, ADCA dramatically decreases the packet delay when the traffic load is below saturation. When the traffic load is below $3000Kbps$, ADCA decreases the delay of MMAC by almost half. Particularly, in the interval of $[1800Kbps, 3000Kbps]$, ADCA achieves dramatically lower delay among all three protocols. The larger delay of 802.11 is because of the long waiting time in queue, while the larger delay of MMAC is because it only allows packets to be delivered one hop in each time interval. At loads larger than $3000Kbps$, the improvement of ADCA over MMAC is less dramatic, because the network is getting saturated. The queue length is over QT most of the time, and thus ADCA does not further negotiate channels and behaves similar to MMAC. From the simulation results, we can conclude that ADCA is able to achieve lower delay than MMAC without impacting the network throughput.

2. Impact of Number of Channels

We repeat our experiments with 6 orthogonal channels. The simulation results are shown in Fig.5.8. In Fig.5.8(a), the maximum throughput of MMAC and ADCA is over 3 times of 802.11. This capacity improvement is higher than the previous exper-



(a) Network Throughput (b) Packet Delay

Figure 5.8: Performance of ADCA under 50 Nodes and 6 Channels

iment, because the use of more orthogonal channels enables more links to transmit and receive simultaneously. Note that the capacity of MMAC and ADCA does not increase linearly (i.e., it is not doubled by doubling the number of channels), because the throughput depends on both the number of channels and the network topology (or the density of links with data transmission demands within an interference range).

Fig.5.8(b) illustrates the average packet delay of the three protocols, which shows the same trend as the case of 3 channels. When the traffic load is below $3700Kbps$, ADCA decreases the delay dramatically compared with MMAC. Especially when the load is in $[1800Kbps, 3700Kbps]$, ADCA achieves the lowest delay among the three protocols. One significant difference is that the load interval at which ADCA excels becomes larger than the case of 3 channels, which means ADCA shows sizable improvement when more channels are used.

To describe this trend in a general way, let T_1 be the saturation point of the single channel MAC and T_m be the saturation point of MMAC or ADCA using m channels. ADCA achieves the best performance when the traffic load is between T_1 and T_m. Note that when the load is below T_1, although ADCA still outperforms MMAC, the single channel MAC protocol works best. This is because the traffic in the network

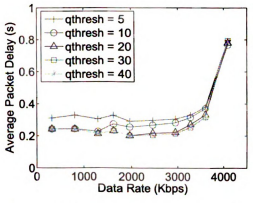


Figure 5.9: Impact of Queue Threshold on Packet Delay in ADCA

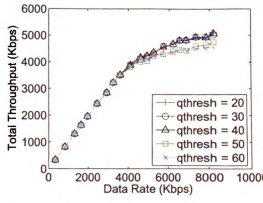


Figure 5.10: Impact of Queue Threshold on Network Throughput in ADCA

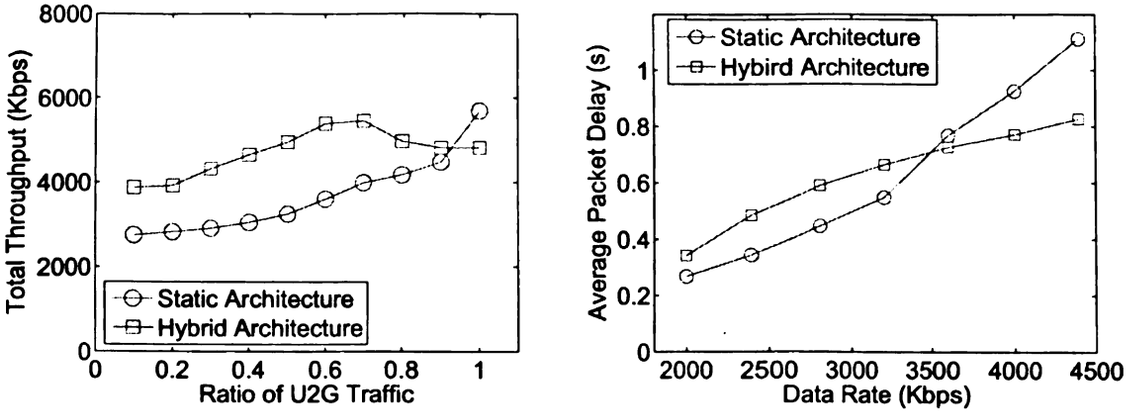
is so light that there is no need to use multiple channels.

3. Impact of Queue Threshold on ADCA

The parameter QT in ADCA is the threshold that controls whether to enable further channel negotiation on each node. In order to analyze its impact on the protocol, we repeat the simulation on 50 node topology with 6 channels by varying the values of QT. The simulation results are shown in Fig.5.9 and Fig.5.10. We can observe that the packet delay decreases with the increasing threshold. However, when QT is between 20 and 40, the performance of ADCA becomes stable. Through experiments, we also found that when QT is over 40, there is no dramatic decrease in packet delay, but the network throughput degrades gradually compared to MMAC.

As can be seen from the simulation results, ADCA achieves best performance when QT is between 20 and 40. When QT is too small, the channel negotiation occurs between only a pair of nodes in most cases even though the traffic load is far below the saturation point of multi-channel networks. On the other hand, if QT is too large, with the increasing traffic load, queues easily get overloaded, which causes packet loss and even increase in packet delay.

We repeat this analysis on networks with varying number of nodes and different



(a) Network Throughput

(b) Packet Delay (U2G Ratio = 0.5)

Figure 5.11: 50 nodes with 6 channels under Mixed Traffic

number of channels, and find that the optimal interval for QT is quite stable, not affected with network configurations dramatically. Thus, we set QT to 30 in our protocol.

5.5.2 Compare Hybrid Architecture with Static Architecture

In this subsection, we evaluate the throughput of the hybrid network, and compare it with the pure static architecture. We consider a wireless mesh network with 50 nodes randomly deployed in a $1200m \times 1200m$ area. There is one node designated as the gateway node in the middle area. Each node has 3 interfaces. In the pure static network, all the interfaces of each node work with static channels, while in the hybrid network, 2 interfaces of each node work with static channels and the other interface works with dynamic channels. In the simulation, we use interference and congestion aware routing algorithm on the hybrid architecture and use ETX routing on the static architecture. Evaluations are performed under different traffic patterns.

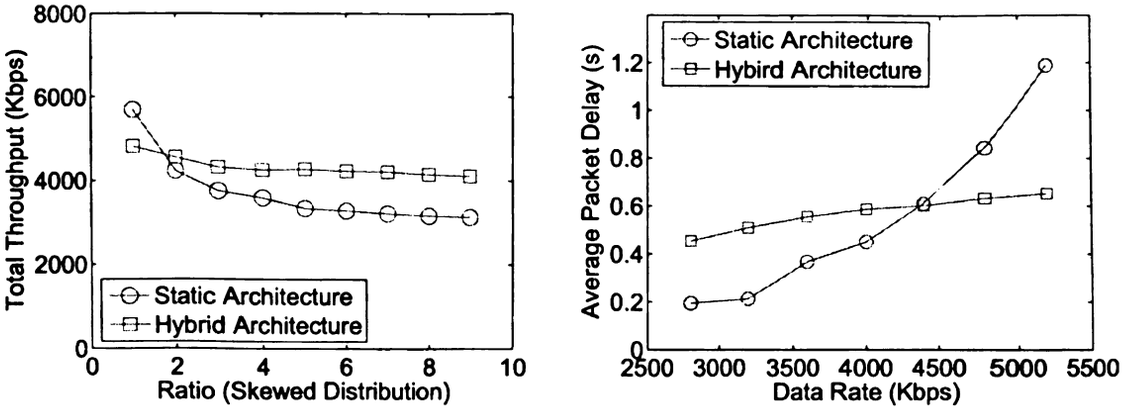
1. Mixture of $U2G$ and $P2P$ Traffic

In this setting, we consider two types of traffic in the network: 1) $U2G$, connections between end-users (non-gateway nodes) and the gateway; 2) $P2P$, connections between end-users. In both types, end-user nodes are selected randomly. In our experiment, we generate 50 random flows, each with the same data rate, and vary the ratio of the number of $U2G$ flows over all flows ($U2G$ and $P2P$). The maximum network throughput is measured by the total throughput of all flows when the network is getting saturated by the flows.

Fig.5.11 illustrates the simulation results with 6 orthogonal channels. As shown in Fig.5.11(a), the static architecture works best when the ratio is 1 (there is only $U2G$ traffic). With the decreasing ratio, the throughput of the static network decreases dramatically, because it is not able to provide good routing paths for $P2P$ traffic in most cases. In contrast, there is no dramatic throughput degradation for hybrid network with the changing ratio. We can observe that the hybrid architecture always outperforms the static architecture except when the ratio is close to 1, because the channel allocation of the static network is especially optimized for this particular case. Fig.5.11(b) shows the average packet delay when the ratio of U2G traffic is 0.5. When the traffic is low, the hybrid architecture has higher delay, which is mainly caused by dynamic channel switching. However, when the traffic becomes heavy, the static architecture tends to have higher delay than the hybrid architecture, because the traffic load reaches the saturation point for the static architecture while it does not for the hybrid architecture.

2. Skewed Traffic Distribution

In this setting, all the traffic is of $U2G$ type, but we make the traffic distribution skewed by controlling the probability of flow generation for each node. For example, we divide the network into two areas evenly, and let each node in one area have



(a) Network Throughput

(b) Packet Delay (Skewed Ratio = 5)

Figure 5.12: 50 nodes with 6 channels under Skewed Traffic

probability of p_l to generate a flow to the gateway, and each node have probability of p_r in the other area. We vary the ratio p_l/p_r and generate 50 random flows in our experiment.

Fig.5.12 illustrates the simulation results with 6 orthogonal channels. In Fig.5.12(a), the static network works best when the ratio is 1 (every node has equal possibility to generate a flow to the gateway). However, with the increasing ratio, the throughput of the static network decreases dramatically. In contrast, the hybrid network is better at dealing with nonuniform traffic. There is no dramatic throughput degradation with the increasing ratio. Fig.5.12(b) shows the average packet delay when the ratio of 5. When the traffic is low, the hybrid architecture has higher delay, because some flows have been routed via a detoured path across the less congested region through dynamic links. However, when the traffic becomes heavy, the hybrid architecture achieves lower delay than the static architecture, because it has high network capacity.

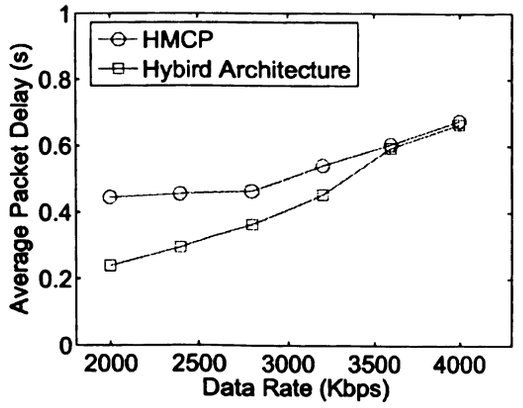


Figure 5.13: 50 nodes with 8 channels under Mixed Traffic

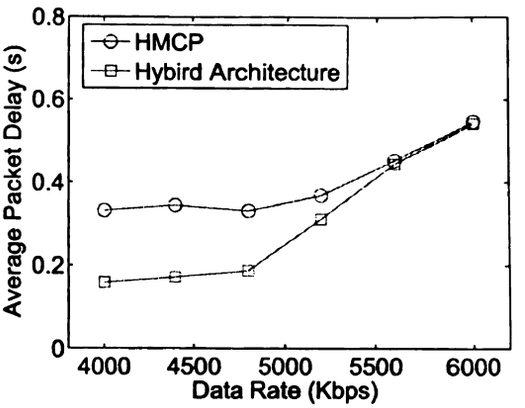


Figure 5.14: 50 nodes with 8 channels under Skewed Traffic

5.5.3 Compare Hybrid Architecture with HMCP

In this subsection, we compare our hybrid architecture with HMCP [59]. In HMCP, for each node, some interfaces are assigned with fixed channels, while the remaining interfaces can frequently switch channels. Each node informs its neighbors of its fixed channels by broadcasting a message on all the channels to its neighbors. If a node needs to transmit some packets to a neighbor, it will first switch one of its switchable interfaces to a channel, which one of the neighbor's fixed interfaces is working on, and then send the packets through the switchable interface. Although this approach also maintains good adaptivity to changing traffic, it causes a high delay for multihop data transmission, because the data transmission on each hop needs a channel switch. We simulate HMCP on the same topology setting. In this case, each node is equipped with 3 interfaces, where two of them are dynamic interfaces and the other one is static interface. Each dynamic interface switches channels every $100ms$, which is the same with our algorithm.

We simulate both approaches on 50 node topology with 8 channels. The comparison between HMCP and our algorithm is illustrated in Fig.5.13 and Fig.5.14. We can observe that HMCP causes high delay even under low traffic. However, the packet delivery delay has been decreased dramatically by our approach. When the traffic

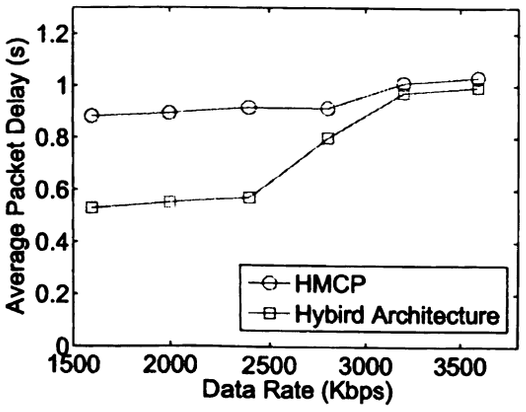


Figure 5.15: 100 nodes with 8 channels under Mixed Traffic

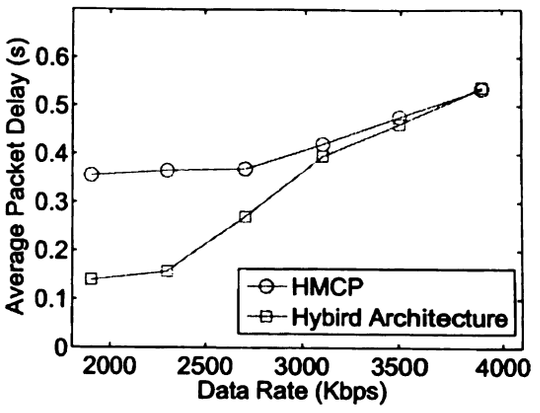


Figure 5.16: 100 nodes with 8 channels under Skewed Traffic

load becomes heavy, the delay of both approaches increases to the similar level. The high delay is due to the long waiting time in queues for both approaches, which becomes a major part of the overall delay with regard to the time spent on channel switching. We also simulate both approaches on 100 node topology with 8 channels, and the results are shown in Fig.5.15 and Fig.5.16. We can observe the same trend in this larger scale topology.

5.6 Summary

In this chapter, we have proposed a hybrid wireless mesh network architecture, where each mesh node has both static and dynamic interfaces. We have made two contributions. First, we proposed an adaptive dynamic channel allocation protocol (ADCA) to be used on dynamic interfaces. Compared with MMAC, ADCA reduces the packet delivery delay without degrading the network throughput. In addition, we proposed an interference and congestion aware routing algorithm in the hybrid network, which balances the channel usage in the network and therefore increases the network throughput. The simulation results have shown that, compared to the pure static architecture, our approach increases the overall network throughput by around

20 to 30 percent under mixed or skewed traffic, without causing obvious increase in overhead. Moreover, compared to existing hybrid architecture, our approach is able to decrease the delay by around 50 percent under low data traffic, while maintaining the adaptivity to the changing traffic.

CHAPTER 6

Multi-Path Routing for Video Streaming in Wireless Mesh Networks

The video on-demand application (VoD) has become a popular Internet service recently. There have already been several commercial products developed to support VoD applications, such as PPLive and PPStream. Most of them use peer-to-peer ($P2P$) technology to improve the VoD performance. Such architecture has been discussed in [104]. Assume users can store the videos that they have recently watched in their local storage (e.g., PPLive and PPStream buffers $1G$ bytes of the most recently watched videos, which is enough for over two 2-hour movies, in a peer's local storage). When a client wants to watch a new video, he or she first discovers which peer clients have buffered the video, and then streams the video from both the servers and peer clients through multiple paths. The multi-path multi-source video on-demand streaming has been applied in wired networks with great success. However, it remains a challenging task in wireless networks due to the effect of wireless interference.

A community wireless mesh network is a static multi-hop wireless network com-

posed of many mesh routers, where each mesh router establishes connectivity with neighboring mesh routers. There are some special routers working as gateways to provide access to the Internet. In a large community network, when a VoD user initiates a video request, it can stream the video from two types of sources: 1) The servers or peers that have buffered the video within the community network; 2) Even if there are no such sources, the user can stream the video from multiple servers or peers in the Internet through the multiple gateways. As VoD applications have high demand of bit rate, delay, and loss sensitivity, the bottleneck of the multi-path video streaming is usually in the community network, that is, the multi-hop wireless paths from the user to peers or servers in the community network or the paths from the user to gateways. Fortunately, in multi-channel multi-radio wireless mesh networks, the enhanced channel diversity increases the network capacity. In this chapter, we study multi-path routing and rate allocation in multi-channel multi-radio wireless mesh networks to improve the VoD performance.

One major problem for the multi-source VoD application in wireless mesh networks is the discovery of multiple independent paths. By splitting the video stream over multiple independent paths, we can not only improve the aggregate routing performance, but also improve the stability and robustness. In the Internet, the independent paths are usually defined as edge-disjoint or vertex-disjoint paths. In edge-disjoint paths, no two paths share a same link, and therefore any link failure will only affect one path. Vertex-disjointness is stronger than edge disjointness, because it also guarantees that any node failure will affect at most one path. In wireless mesh networks, the discovery of independent paths becomes more challenging due to wireless interference. Even if two paths are edge-disjoint or vertex-disjoint, they might still affect each other if they have wireless links that interfere with each other. Thus, their aggregate routing performance becomes lower than expected, and the congestion of one path will probably influence the other path. This route coupling

effect has been studied in [80]. Therefore, in order to find independent paths in wireless mesh networks, we should guarantee interference-disjointness in additional to edge-disjointness or vertex-disjointness.

Another challenge is that the optimization of VoD performance should be considered over all VoD users in the network instead of only the current user. As we know, the wireless mesh network is designed to be shared among multiple users. If the routes with required bandwidth have been found for a VoD request, the VoD user can establish connections in the network for data transport, which we call a VoD session. There might be multiple coexisting VoD sessions from different users in the wireless mesh network. Thus, we should not be too selfish when finding the routes and rate allocation for the current VoD request. Instead, we should consider not only the resulting performance of the current VoD session, but also its influence on the other existing VoD sessions in the network and the network's ability of satisfying new VoD requests.

In this chapter, we consider the wireless interference in the discovery of multiple independent paths. For each arriving VoD request, given n senders (servers or peers), which can provide the same video, we find the maximum number of m ($m \leq n$) high-quality and independent paths that connect from m senders to the receiver (the new user who initiates the VoD request). We formulate it as a constrained maximum independent paths problem, and design efficient path discovery algorithms. Based on the multiple paths discovered, we further propose a joint multi-path routing and rate allocation algorithm to determine the routes together with the rate allocated for each route for the VoD request, with the goal of minimizing the network congestion. By using this optimization framework, we not only improve the average performance of existing VoD sessions, but also enable the network to support more subsequent VoD requests. We evaluate our algorithms by using video trace simulations.

The rest of chapter is organized as follows. In Section 6.1, we present the system

model and problem formulation. In Section 6.2, we describe two proposed heuristic algorithms for multi-path discovery. In Section 6.3, we propose a joint routing and rate allocation algorithm. The simulation results are shown in Section 6.4. We summarize the chapter in Section 6.5.

6.1 System Model and Problem Formulation

In this section, we first introduce the system model, and then formulate the problem of multipath discovery.

6.1.1 Multi-Source VoD in WMN

Consider the VoD application in a large community network constructed by wireless mesh networking technology. The network is composed of a number of multi-channel multi-interface wireless mesh routers. Each mesh router can establish wireless connectivity with neighboring mesh routers, so that a static multi-hop wireless network is formed. There are some special mesh routers working as gateways, which are directly connected to the Internet. Previous static channel allocation algorithms, such as [50] [99], can be used to assign each interface of each router with a channel so as to minimize the network interference while maintaining the network connectivity.

For some popular videos, the VoD performance can be improved by $P2P$ technology. Whenever a user has requested a video, it registers to the server, so that the server keeps a list of the users that have buffered the video. When a new user visits, he or she queries the server for the list of users, from which they can stream the video. If there are such users, the new user can stream the video not only from the media server, but also from the other peers. For the peers that are located in the community network, the user downloads over a multi-hop wireless path, while for the peers in the Internet, the user downloads over a path, which consists of a multi-hop

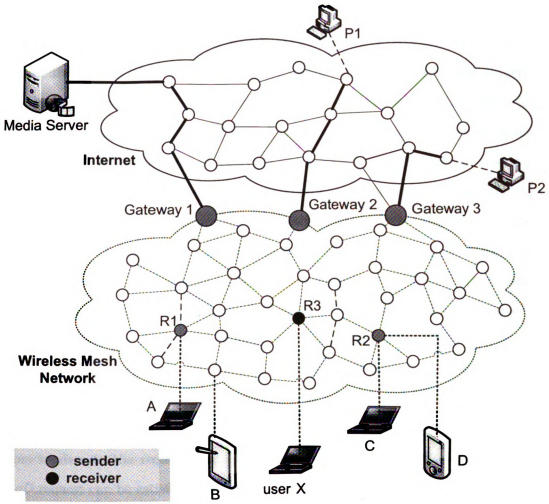


Figure 6.1: VoD in Wireless Mesh Networks

wireless path from the user to one of the gateways and a path from the gateway to the peer in the Internet.

We call the peer or the server that provides the download of the video as the *sending node*, and the user that requests the video as the *receiving node*. A mesh router is called a ***sender*** (or ***receiver***) if it is connected with one or more sending nodes (or the receiving node). If the sending node is located in the community network, the mesh router that the user is directly connected with is the sender. If the sending node is in the Internet, the gateway through which users in the community

network can access it is the sender. In the example illustrated in Fig.6.1, where there is a media server in the Internet, we assume $P1$, $P2$, A, C are the peers that have buffered the video. If X want to watch the video, there are five available senders in this case, which include $R1$, $R2$, and the three gateways. Note that it is possible for a mesh router to be both a sender and a receiver if it connects with both some sending nodes and the receiving node. In this case, the receiving node can always stream the video from these sending nodes through their common mesh router, because the video transmission is not contending with other router-to-router links. Besides, we can still utilize multiple paths from the other senders to the receiver for the bandwidth that cannot be satisfied by streaming from local senders only. Therefore, it is sufficient to consider the case where a mesh router will not be both a sender and a receiver at the same time in the problem formulation.

In this chapter, we focus on the problem of multipath routing and rate allocation to improve the video streaming performance in multi-channel multi-radio wireless mesh networks. For each arriving VoD request, we need to determine the routes from senders to the receiver and the rate on each route for video streaming. A VoD request can be satisfied if we find the routes with the required bandwidth that convey the demanded video quality; otherwise, the VoD request cannot be satisfied, and it will be blocked. If a VoD request can be satisfied, the receiver will establish connections for video streaming in the network. For a VoD request, from the time when the connections are established to the time when they are closed due to the end of video streaming, we call the set of connections as a ***VoD session***. In the following, we will formulate the problem of multipath discovery for each VoD request.

6.1.2 Multipath Discovery

Consider a wireless mesh network $G(V, E)$, where V denotes the set of mesh routers whose locations are known. There is an undirected edge $(u, v) \in E$ if and only if

$d(u,v) \leq R$, where $d(u,v)$ is the Euclidean distance between mesh routers u and v, and R denotes the maximum wireless radio transmission range.

Suppose each mesh router is equipped with Q interfaces, and there are K orthogonal channels available. Given a channel assignment A, where $A(u)$ $(u \in V)$ denotes the set of channels assigned to the interfaces of u, the *topology* $G_A(V, E_A)$ of the network can then be determined. There is a wireless link $e = (u, v, c) \in E_A$ if and only if $(u,v) \in E$ and $c \in A(u) \bigcap A(v)$, that is, u and v each have at least one interface working on channel c.

We use the protocol model to determine whether two wireless links in G_A interfere with each other or not. For any two links $e_i, e_j \in E_A$, define their distance $d(e_i, e_j)$ as the minimum Euclidean distance between any node of one link and any node of the other link. e_i and e_j interfere with each other if: 1) $c(e_i) = c(e_j)$, where $c(e)$ denotes the channel of wireless link e. 2) $d(e_i, e_j) \leq I$, where I is the wireless interference range, which is usually $2R$.

Definition 1: Given topology $G_A(V, E_A)$, the ***conflicting table*** is the set of all link pairs (e_i, e_j), where $e_i, e_j \in E_A$ and $e_i \neq e_j$, such that e_i and e_j interfere with each other.

In the topology $G_A(V, E_A)$, let $r \in V$ be the receiver, and assume there are n senders $S = \{s_1, s_2, ..., s_n\} \subseteq V$. We consider the case where a mesh router will not be a sender and a receiver at the same time, that is, $r \notin S$.

Definition 2: Given topology G_A and its conflicting table T, two paths p and q interfere with each other if there exists $(e_i, e_j) \in T$ such that $e_i \in p$ and $e_j \in q$. The ***Maximum Interference Disjoint Paths problem (MIDP)*** seeks the maximum number of edge disjoint paths from S to r, denoted by P, where each path is between a different sender in S and r, such that there does not exist (p_i, p_j), where $p_i, p_j \in P$ and p_i p_j interfere with each other.

In other words, the problem finds the maximum number of paths from the set of

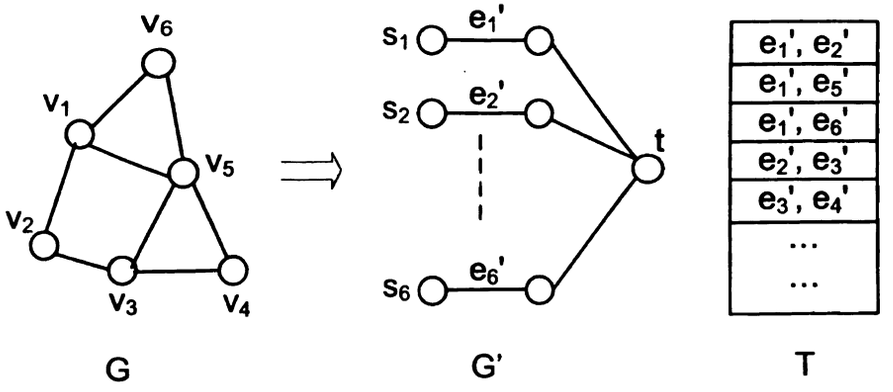


Figure 6.2: Problem Reduction

senders to the receiver, such that any two paths are independent from each other with regard to wireless interference. Let $n =\mid S \mid$ be the number of senders and $m =\mid P \mid$ be the number of paths, we have $m \leq n$.

Theorem 1: The $MIDP$ problem is NP-hard.

Proof: We can prove this by reducing the maximum independent set problem, which is NP-Complete, to the formulated problem. Given an undirected graph $G(V, E)$, the maximum independent set problem finds the largest independent set (no two nodes from the set are adjacent) in the graph. We first transform the graph G to G' as follows. 1) For each $v_i \in V$, create an edge e'_i in G'. Denote one end vertex of e'_i by s_i, and create an edge from the other end vertex to a vertex t. Fig.6.2 shows an example of the transformation. 2) Create a list T, which is initialized to empty. For each $e = (v_i, v_j) \in E$, add (e'_i, e'_j) to T. As a result, the maximum independent set problem can be reduced to the problem of finding maximum interference disjoint paths from $S = \{s_1, s_2, ..., s_{|V|}\}$ to t in G', given the conflicting table T. The reduction only takes polynomial time. Therefore, the formulated problem is NP-hard.

In the real-world VoD applications, we not only want to find more interference disjoint paths from S to r, but also need to guarantee the quality of each path with regard to packet loss, delay, and throughput. There have been many studies on

metrics for finding good routes between a single source and a single destination in wireless networks. The $WCETT$ metric [37] is widely used in multi-channel multi-interface wireless mesh networks. It not only considers packet loss and delay, but also accounts for channel diversity in each path so as to reduce intra-flow interference. Therefore, we use this metric to evaluate the quality of each selected path. More specifically, we want to find the maximum number of independent paths, while keeping the $WCETT$ value of each path below a certain threshold.

Another problem in real applications is that there might not be enough completely interference disjoint paths in some cases, because of the limited number of channels and number of interfaces. To take advantage of multipath streaming, we can relax the constraint on the path independency a little bit to allow more paths to be found, which improves the robustness of applications.

Definition 3: Given a set of edge disjoint paths P in topology G_A, consider any link $e \in p_0$ ($p_0 \in P$). The set of paths that interfere with e is $I(e) = \{p \mid p \in (P - p_0) \bigwedge p$ has a link that interfere with $e\}$. The ***path interference*** of link e is defined as $PI(e) =\mid I(e) \mid$, which reflects how many other paths in P are interfering with this link. The ***maximum path interference*** of P is defined as $MPI(P) = Max_{p \in P} Max_{e \in p} PI(e)$.

Now we can formally describe the problem by considering both the constraint and relaxation.

Definition 4: Given topology G_A and its conflicting table T, the ***COnstrained Maximum INdependent Paths problem (COMINP)*** seeks the maximum number of edge disjoint paths from S to r, denoted by P, such that 1) $MPI(P) \leq \alpha$, where α is the threshold to control the level of independency between paths in P; 2) $WCETT(p) \leq \beta$ for $p \in P$, where β is the threshold to control the quality of each path.

Note that by setting $\alpha = 0$ in the first constraint and $\beta = +\infty$ in the second

constraint, the $COMINP$ problem becomes exactly $MIDP$ problem. Therefore, $COMINP$ is also NP hard.

6.2 Multipath Discovery Algorithm

In this section, we propose two heuristic algorithms, the Iterative Path Discovery algorithm (IPD) and the Parallel Path Discovery algorithm (PPD). We list the definitions of some frequently used notations in Table 6.1. Some notations will be explained in detail when we present the algorithms. Both algorithms run in a centralized fashion on the receiver.

In wireless mesh networks, the mesh routers usually have minimal mobility. For example, in community networks, the routers are usually fixed on roofs of houses. In addition, due to the overhead of dynamic channel switching, static channel allocation strategies are widely used, in which the channel assignment does not change often. This makes it possible for each mesh router to collect the global knowledge of the network, including each other router's position and channel assignment. This can be done by letting each router broadcast the information to the whole network each time the network topology has been reconstructed or channels have been reassigned, which occurs very rarely. Therefore, each mesh router knows the global topology of the wireless mesh network (wireless links and channels), which makes it possible to use a centralized algorithm on the receiver to find multiple paths from senders to the receiver.

6.2.1 Iterative Path Discovery

The Iterative Path Discovery algorithm (IPD) finds paths one by one from the senders to the receiver. In each iteration, we find one path from a sender to the receiver, and then update the topology accordingly. This process continues until no new paths can

Table 6.1: Notations

$T(V,E)$	the original network topology
$T'(V,E')$	the remaining network topology
S	the set of senders
r	the receiver
$IF_T(p)$	total interference of path p in topology T
$IE_T(p)$	path interfering set of p in topology T
$l(e)$	the number of paths interfering with e
$WCETT(p)$	the $WCETT$ value of path p

be found from the remaining topology. There are two critical steps in the algorithm, which we will explain below.

1. Path Selection

Let S' be the set of remaining senders, for which we have not found paths yet, and T' be the remaining topology. Initially, $S' = S$ and $T' = T$. In this step, for each $s \in S'$, we first find a minimum $WCETT$ path p from s to r in T' if such a path exists and $WCETT(p) \leq \gamma \cdot w_T(s,r)$, where $w_T(s,r)$ is the $WCETT$ value of the optimal path from s to r in T and γ is a constant to control the quality of each path (we set $\gamma = 1.5$ in our experiment). Denote the resulting set of paths as P, we then need to decide which path to select from P. As the algorithm aims at discovering as many paths as possible in the final solution, we select a path from P based on the following metric.

We define the *total interference of p in $T'(V,E')$*, denoted by $IF_{T'}(p)$, as the number of edges in T' that interfere with any edge in p. More formally,

$$IF_{T'}(p) = \mid \{e' \mid e' \in E' \bigwedge \exists e \in p : Interfere(e,e')\} \mid$$

Note that $Interfere(e,e')$ is true if $e = e'$.

Therefore, we select the path from P, which has the minimum total interference in T'. The reason is that the selected path will render minimum change in the remaining topology, and thus leave us more flexibility in finding more paths afterwards.

Algorithm 5 Iterative Path Discovery Algorithm

1: $T'(V, E') = T(V, E)$, $S' = S$
2: $l(e) = 0$ for $e \in E'$, $X = \{\}$
3: **repeat**
4: For each $s_i \in S'$, find a minimum $WCETT$ path p_i from s_i to r in T' if $WCETT(p) \leq \gamma \cdot w_T(s, r)$. Denote the set of paths by P.
5: Select p^* from P, which has the minimum total interference in T', $IF_{T'}(p^*) = Min_{p_i \in P} IF_{T'}(p_i)$.
6: $l(e) = l(e) + 1$ for $e \in IE_{T'}(p^*)$
7: $T' = T' - p^*$
If $l(e) > \alpha$, $T' = T' - e$, for $e \in IE_{T'}(p^*)$
8: $S' = S' - sender(p^*)$, $X = \{X, p^*\}$,
where $sender(p^*)$ is the sender of path p^*
9: **until** P is empty
10: X is the set of selected paths.

2. Topology Update

Once a path p has been selected from T', we need to update T' accordingly. We can guarantee the edge-disjointness of paths by taking off p from T', so that the paths found later will not overlap with the paths previously found. In addition, we need to consider p's interference on T', and guarantee the level of independency among the final set of selected paths.

We use a label l on the edges of T' to record the interference from the already selected paths, where $l(e)$ counts how many selected paths are interfering with link e. At the start of the algorithm, $l(e)$ is initialized to 0. Assume p is the selected path from $T'(V, E')$ in the current iteration. We define the *path interfering set of p in T'*, denoted by $IE_{T'}(p)$, as the set of edges in $(T' - p)$ that interfere with any edge in p.

$$IE_{T'}(p) = \{e' \mid e' \in (E' - p) \bigwedge \exists e \in p : Interfere(e, e')\}$$

We update $l(e)$ for each edge $e \in IE_{T'}(p)$ by increasing 1, indicating that there is one more selected path p that interferes with e. If $l(e) > \alpha$, then we need to take off e from T'. This is because if e has been used in one more path in the remaining

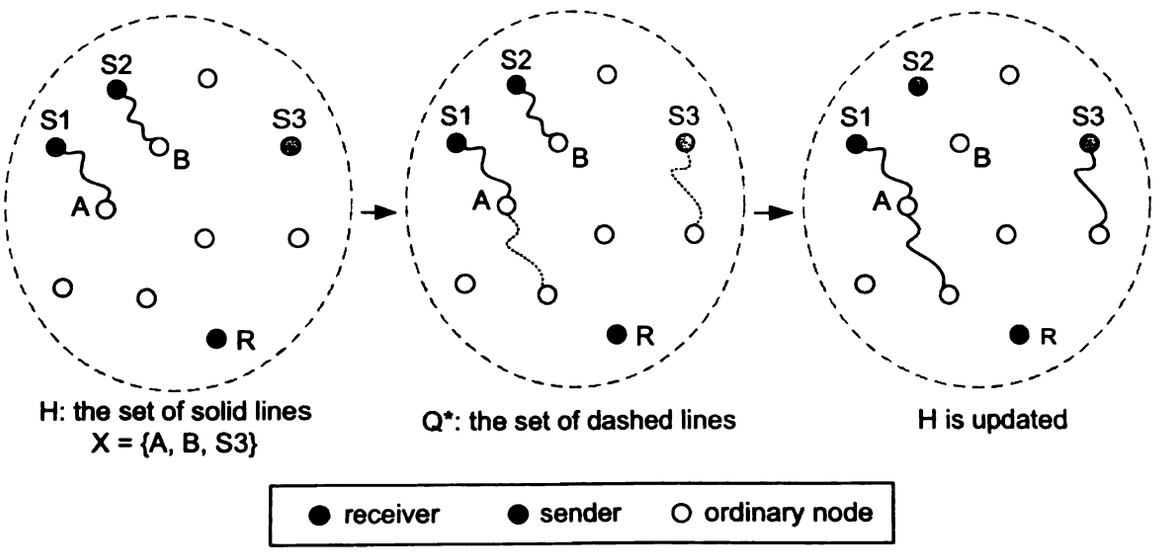


Figure 6.3: The Basic Idea of Parallel Path Discovery

topology, then e will interfere with more than α already selected paths, which violates the constraint in the $COMINP$ problem.

The algorithm is formally described in Algorithm 5. In each iteration, we select a path and update the topology, which takes $O(\mid S \mid\mid E \mid\mid V \mid)$. There are at most $\mid S \mid$ iterations, so the algorithm takes polynomial running time $O(\mid S \mid^2 \mid E \mid\mid V \mid)$.

6.2.2 Parallel Path Discovery

IPD finds a complete path in each step, which limits the search space, and thus may not be able to approximate the optimal solution. Instead of searching for paths one by one, we take a different approach in this section, that is, we perform a parallel search for paths from all senders to the receiver level by level. The basic idea of the Parallel Path Discovery algorithm (PPD) is as follows. In each iteration, we perform a breadth first search from r in the remaining topology $T'(V, E')$, and categorize all the nodes in V into layers based on their distances from r. We start from the senders that are in the farthest layer, and find partial paths that connect as many of them as possible to nodes in the lower layer. We then use these nodes that are reached

in the lower layer as new senders to take place of the original senders. The process continues until we reach the receiver.

1. Parallel Search

As illustrated in Fig.6.3, let $H = \{p_1, p_2, ..., p_l\}$ be the set of partial paths found in the current iteration. Each partial path $p_i \in H$ starts from a sender in S, denoted by $s(p_i)$ and ends at a non-receiver node, denoted by $e(p_i)$. We do a breadth first search on the current remaining topology T'. Let $d(v)(v \in V)$ be the distance from v to the receiver r in T'. Consider the set of nodes $X = \{e(p_i)\} + (S - \{s(p_i)\})$, which contains the set of ending nodes of the partial paths and the set of senders that do not have partial paths yet. These are the nodes from which we can further extend the set of partial paths.

In each iteration, we extend partial paths only from the highest layer nodes in X, that is, $X' = \{v \mid d(v) = d_{max}, v \in X\}$ where $d_{max} = Max_{v \in X} d(v)$. We want to find as many paths as possible that connect the nodes from X' to the next lower layer nodes $Y = \{v \mid d(v) = d_{max} - 1\}$. Assume Q^* is the set of paths found to extend H. Let $H_{Q^*} \subseteq H$ be the set of partial paths that can be extended by paths in Q^*. We extend these partial paths and make them as the new H. In addition, we also need to update T'. We take off paths in Q^* together with their interfering edges whose l values exceed the threshold from T', and also recovering paths in $(H - H_{Q^*})$ back into T'. After we have updated H and T', we do a breadth first search again on the updated topology and continue to extend H following the same procedure.

The algorithm is described in Algorithm 6. There is a key sub-algorithm *Layer-PathSearch* used to extend the set of partial paths to the next lower layer, which we will explain in the following.

Algorithm 6 Parallel Path Discovery Algorithm

1: $T' = T$, $H = \{\}$
2: **repeat**
3: Do a breadth first search on T'
4: $Q^* = LayerPathSearch(H, T')$
5: Extend H and update T' based on Q^*
6: **until** H reaches the receiver r
7: Output all the paths in H.

2. Layer Path Search

We first enumerate all the paths of at most 3 hops long from X' to Y in T'. The reason for doing this is that: 1) If we only use one hop path to extend the partial paths, we are always finding the shortest paths. However, in multi-channel wireless mesh networks, the channel diversity is also important for path quality in addition to hop distance. Therefore, we also check 2 or 3 hop paths in order to utilize the channel diversity in the path selection. 2) If we do not limit the path length, we will have exponential search space. In practice, we find that by keeping the threshold at 3, we have enough flexibility in finding channel diverse paths.

Denote the set of paths by Q; we want to find as many paths as possible from Q to extend H without breaking the edge-disjointness and path independency constraints. We rate each path $q \in Q$ based on two factors. 1) The interference caused by q on T', denoted by $IF_{T'}(q)$. The path that causes less interference is preferred to be selected, because it increases the possibility of finding other more paths later. 2) The interference caused by q on the connecting partial path $p \in H$, that is, q starts from the ending node of p. We denote it by $IF_p(q)$. This is related with the channel diversity of each final path discovered. While we want to find more paths, we also need to guarantee the quality of each path. Therefore, q can be evaluated by $z(q) = k_1 \times IF_{T'}(q) + k_2 \times IF_p(q)$.

The algorithm is described in Algorithm 7. We take the path with the lowest z value from Q each time. When we have selected a path $q \in Q$ to extend a partial

Algorithm 7 FindPathSet(H, T')

1: $Q^* = \{\}$
2: In T', find each path q that is at most 3 hops from X' to Y such that $WCETT(p+q) \leq \gamma \cdot w_T(sender(p), r)$, where $p \in H$ concatenates with q. Denote the set by Q.
3: Rate each path $q \in Q$ by metric $z(q)$.
4: **repeat**
5: Take the path q^* with the lowest z value in Q.
6: $Q^* = Q^* + q^*$
7: Update T' and Q based on q^*.
8: **until** no paths can be selected
9: Q^* is the set of paths to extend H

Algorithm 8 LayerPathSearch(H, T')

1: $H' = H$
2: $Q^* = FindPathSet(H', T')$
3: **repeat**
4: For each $p_i \in H'$, release p_i and update T', $Q_i = FindPathSet(H' - p_i, T')$
5: Let Q_k be the best solution among $\{Q_i\}$
6: **if** Q_k is better than Q^* **then**
7: $Q^* = Q_k$, $H' = H' - p_k$
8: **end if**
9: **until** no better solution can be found
10: Q^* is the path set to extend H.

path $p \in H$, we need to update T' in the same way as described in the iterative path discovery algorithm. We must not only take off q from T', but also take off edges whose l value exceeds the threshold α in T'. As some edges have been taken off from T', we also need to update Q, because some paths may become unavailable in the updated topology. We continue to select paths from Q until Q becomes empty. As a result, we get the set of paths to extend H.

Given H in the current iteration, it is possible that H contains too many partial paths so that they are occupying too many link resources in the lower layers due to wireless interference (these links have been removed from the original topology be-

cause their l value exceeds the threshold). This may dramatically reduce the number of paths finally discovered. To deal with this problem, we can release some partial paths from H first, so that we can have enough link resources in the lower layers, and thus we may be able to extend more partial paths to the next layer. The algorithm is described in Algorithm 8, which can be summarized as trying to find a local minimum point.

Let D be the diameter of T. In Algorithm 7, we consider $O(\mid S \mid (\frac{\mid V \mid}{D})^3)$ partial paths, and thus the running time is $O(\mid S \mid (\frac{\mid V \mid}{D})^3 \mid E \mid)$. Algorithm 8 calls Algorithm 7 $O(\mid S \mid^2)$ times, and thus takes $O(\mid S \mid^3 (\frac{\mid V \mid}{D})^3 \mid E \mid)$. There are $O(\mid D \mid)$ iterations in Algorithm 6. Therefore, the algorithm takes polynomial running time $O(\frac{\mid S \mid^3 \mid V \mid^3 \mid E \mid}{\mid D \mid^2})$.

Although the parallel path discovery algorithm takes more computation overhead than the iterative path discovery algorithm, it has the promise of finding more paths. In the iterative path discovery algorithm, we find one complete path each time, and modify the remaining topology based on the path. As a result, we have made a big change on the remaining topology, and soon no new paths could be found. In contrast, the parallel path discovery algorithm finds partial paths in each step, and leaves more flexibility in the remaining topology for finding more paths.

6.2.3 Discussion

If we want to find strictly edge-disjoint paths, the number of paths that can be found will be limited by the number of interfaces that the receiver has. In order to find more paths to enhance the robustness of the application and provide more flexibility for rate allocation, we can relax this constraint by allowing paths to merge at the last hop towards the receiver. Both the iterative and parallel path discovery algorithms can be easily modified to deal with this case. When taking off a selected path from the remaining topology, we do not remove the edge that is directly connected with the receiver. It will be taken off from the remaining topology only when its l value

is over the threshold α. In this way, we have still guaranteed the path independency constraint on the last hop.

6.3 Joint Routing and Rate Allocation

As the wireless mesh network is designed to be shared among multiple users, the routing and rate allocation for each VoD request should not only consider the performance of itself, but also take into account the existing VoD sessions and the network's ability of satisfying more subsequent VoD requests. Note that the performance of each existing VoD session is dramatically affected by the traffic load on each link used by the session. If the traffic load on a link is high, the application may experience high queuing delay and jitter due to the congestion on this link. In addition, the increase in the number of congested links may disrupt the network's connectivity, thereby causing the network to be incapable of finding routes with required bandwidth for new VoD requests. Therefore, we take our optimization goal as minimizing the network congestion.

In this section, we first propose an optimal rate allocation algorithm on the multiple discovered paths, which determines the rate on each path. It is possible that some discovered paths may be unused (or allocated with zero rate), because they are more congested than others. We then propose a joint routing and rate allocation algorithm with the goal of minimizing the network congestion. Unlike previous work [78] [76] [126] that optimize for the performance of a single session only, our optimization not only improves the performance of existing sessions, but also improves the network's capability of satisfying more subsequent VoD requests. The algorithm runs in a centralized fashion on the receiver. We assume each router periodically broadcasts the available bandwidth of its neighboring wireless links to all the routers in the network, so that each router knows the available bandwidth on all the links in the network

(similar to [50]).

Assume the multiple paths discovered for a VoD request is $P = \{p_1, p_2, ..., p_m\}$, the receiver needs to determine the optimal data rate on each path. Let r_k be the data rate on path p_k ($k = 1, 2, ..., m$), and R be the total data rate required by the VoD session. We have

$$\sum_{k=1,...,m} r_k = R \tag{6.1}$$

The traffic load on each link $e_i \in P$, denoted by $t(e_i)$, is

$$t(e_i) = \sum_{p_k \in P \bigwedge e_i \in p_k} r_k \tag{6.2}$$

Let $A(e_i)$ be the available bandwidth on link e_i. Let IE_{ij} denote whether the two links e_i and e_j interfere with each other. $IE_{ij} = 1$ if it is true, and $IE_{ij} = 0$ if otherwise. The residue capacity of each link $e_i \in P$ under the rate allocation $\{r_1, r_2, ..., r_m\}$, denoted by $C(e_i)$, is

$$Z(e_i) = A(e_i) - t(e_i) - \sum_{e_j \in P \bigwedge e_j \neq e_i} t(e_j) \times IE_{ij} \tag{6.3}$$

Therefore, our optimization goal is to maximize the residue capacity of the bottleneck link (or minimize the network congestion).

$$Maximize\ Min_{e_i \in P}\{Z(e_i)\} \tag{6.4}$$

The problem (1)-(4) is a Max-Min Linear Programming problem, and can be transformed to a general Linear Programming (LP) problem. Therefore, we can solve for the optimal rate allocation $\{r_1, r_2, ..., r_m\}$ in polynomial time. Note that it is not mandatory for all the paths in P to be used for the VoD request. It is possible for some paths to be allocated with rate of zero by solving Equation 6.4, because they are more congested than others. In other words, *the multipath discovery algorithm gives candidate paths, while the rate allocation algorithm (by solving Equation 6.4)*

Algorithm 9 Joint Routing and Rate Allocation

1: Let C be the capacity of each link, $A(e)$ be the available bandwidth of link e. Set $Thresh = C$.
2: Generate sub-topology $T[Thresh]$, which only includes links e where $A(e) \geq C - Thresh$.
3: Find paths P for the session on $T[Thresh]$. Calculate optimal rate allocation r on P.
4: Set $P^* = P$, $r^* = r$. Let v^* be the value of the objective function under (P, r) (Equation (4)).
5: **repeat**
6: $Thresh = Thresh/2$.
7: Find paths P for the session on $T[Thresh]$. Calculate optimal rate allocation r on P.
8: Let v be the value of the objective function.
9: **if** $v \geq v^*$ **then**
10: $P^* = P$, $r^* = r$, $v^* = v$
11: **end if**
12: **until** (no valid P or r exists $||$ $v < v^*$)
13: return (P^*, r^*)

determines the subset of paths to be used together with the rate on each path for the VoD request.

To approximate the optimal joint routing and rate allocation, we use binary search in Algorithm 9. We use IPD or PPD for path discovery sub-algorithms. They not only find multiple high-quality and independent paths, but also are better at balancing the traffic in the network. By finding edge-disjoint paths, the traffic can be well distributed over the network spatially. By minimizing the interference among paths, the traffic can be well distributed over different channels. In Algorithm 9, we first apply path discovery (IPD or PPD) and rate allocation (LP) algorithms sequentially on the original topology to find an initial solution. We then truncate the topology based on a threshold in each step, that is, we only keep the links with enough available bandwidth. We apply the same sub-algorithms and find new solutions. The search terminates if we cannot find any better solutions by decreasing the threshold.

After a receiver has calculated the routing and rate allocation for a VoD request,

it needs to reserve the bandwidth on the paths. It is possible that the reservation fails because the bandwidth has already been reserved by some other VoD request arriving at almost the same time. More specifically, when the receiver calculates the routing and rate allocation, its knowledge of the current network status does not get updated of the other VoD session in time. In this case, we can resolve it by letting the receiver wait for a random time and recalculate the rate allocation on the multiple paths already discovered until it succeeds with reserving the bandwidth. For the application in a real wireless mesh network, this case happens rarely, because it takes a short time for a receiver to calculate the routing and rate allocation for a VoD request (less than $0.3s$) and to reserve bandwidth together with updating link status (less than $0.4s$) in a 60 nodes wireless mesh network. According to the statistics of all the VoD requests for a popular VoD server over the Internet [121], the VoD request arrival rate is around one request per second. As a result, we can expect that the mean interarrival time of VoD requests in a regional wireless mesh network should be at least in minutes. For Poisson process (with the mean interarrival time of 1 minute), the probability that two or more requests arrive within $0.7s$ is less than 1.2 percent, which is a very small probability.

In the VoD application, after a receiver has established a VoD session, some senders might become unavailable due to the following reasons. 1) When the peer continues to watch new movies and its buffer is full, the oldest movie will be removed from the buffer. 2) There is a link failure or node failure in the network, which makes the path to some sender disconnected. An intuitive way is to run the algorithm on the network again. However, this will affect the paths that the receiver is currently using. A better way is to use the algorithm to find more paths in the rest of the topology without affecting the existing paths used. However, if the event of any path failure or any sender leaving the network happens more than a certain number of times on the session, it is better to re-run the algorithm for it.

6.4 Performance Evaluation

In this section, we present the simulation results from $NS2$ based on real video traces.

6.4.1 Simulation Methodology

We perform the simulations in $NS2$. The Hyacinth extension [6] has been used to support multiple channels and multiple interfaces per node in the simulator. In all the simulations, we set the maximum radio transmission range to 250 meters and the interference range to 500 meters. We use 802.11 with bit rate of $11Mbps$ for each interface. Unless specifically stated, the simulation is performed in a 60 nodes random topology within an area of $1500m \times 1500m$. Each mesh router is equipped with 4 interfaces, and there are 8 orthogonal channels used in the channel assignment. We use the channel allocation algorithms proposed in [99] to statically assign channels to each interface in order to minimize the wireless interference within the network.

We use the video traces available in the public domain [2] [95] [28] [29]. Each video trace file includes the size of each encoded video frame together with its quality. During $NS2$ simulation, The frames are encapsulated into UDP packets during network transmission and reconstructed at the receiver. Table 6.2 illustrates the three video traces we use for evaluation. Each movie is 30 minutes long. All the traces use GoP (Group of Pictures) length of 16 with I-frame as the first frame in each GoP, and have a rate of 30 frames per second. All the trace files being used contain the frames in the encoder order. In other words, the frames are transmitted through the network in the encoder order.

We compare the following methods of path discovery for multi-source video streaming. 1) $MinW$: find a minimum $WCETT$ path between each sender and the receiver. Thus, if the network is connected and there are n senders, this method will find n paths. 2) $MEDP$: find the maximum number of edge-disjoint paths from

Table 6.2: Video Traces Used in Simulation

Movie Name	Encoding	Quantization Scale	Average Bit Rate (bps)
Silence of the lambs	H.264, Single-Layer	(22, 22, 24)	358365.5
Star Wars IV	H.264, Single-Layer	(22, 22, 24)	373880.4
Die Hard	H.264, Single-Layer	(22, 22, 24)	369614.9

the senders to the receiver (Edge disjoint paths have been used in both Internet [14] and multi-hop wireless networks [61]). 3) IPD: use the iterative path discovery algorithm to find the maximum number of independent paths. 4) PPD: use the parallel path discovery algorithm, which has the potential to find more paths than IPD under the same constraints. To be consistent in the comparison, we use the joint routing and rate allocation framework (Algorithm 9) for all the methods. In other words, we use Algorithm 9 with different path discovery algorithms as sub-algorithms to find multiple paths, while using the same rate allocation algorithm for all the methods.

We evaluate different methods based on the following metrics in our simulation.

- The maximum number of concurrent VoD sessions that can be supported in the network. This metric reflects the network capacity of supporting VoD applications.

- Packet delivery delay. This metric determines the response time of the VoD application. As we are streaming data over multiple paths, each path may possess a different average packet delay. In this case, we calculate the maximum average packet delay over all paths.

- Packet delay jitter. This metric is critical for the quality of video streaming. For a single path, we can evaluate delay jitter by the variance of delays. Let $d_i (i = 1, 2, ..., k)$ be the delay of each packet in a single path within an interval of time, the delay jitter of $d = \{d_i \mid i = 1, 2, ..., k\}$ can be evaluated by $jitter(d) = \sqrt{\sum_i (d_i - \bar{d})^2 / k}$, where $\bar{d} = \sum_i d_i / k$. We can extend this metric to multiple

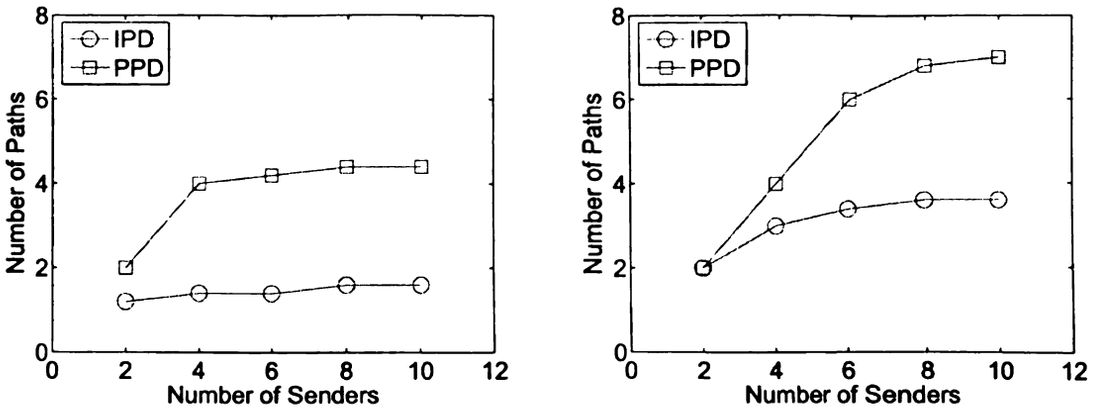
paths in the following way. Let $d_{ij}(i = 1, 2, ..., m; j = 1, 2, ..., k_i)$ be the delay of the jth packet in the ith path. The delay jitter of $d = \{d_{ij} \mid i = 1, 2, ..., m; j = 1, 2, ..., k_i\}$ can be evaluated by $jitter(d) = \sqrt{\sum_i \sum_j (d_{ij} - \bar{d}_i)^2 / \sum_i k_i}$, where $\bar{d}_i = \sum_j d_{ij}/k_i$.

- Packet drop ratio. When a receiver streams a video over multiple paths, it needs to determine a playout deadline, that is, the time the receiver waits for before playing out the video. As a result, the deadline of each packet to be received can be determined. A packet drop occurs when 1) the packet is lost due to the collision in wireless transmission; 2) the packet has been successfully received, but it is received after its deadline. The packet drop ratio is defined as the number of packets dropped over the total number of packets transmitted. It influences the quality of the decoded video.

- Percentage of frames reconstructed. Note that there are dependencies between the encoded video frames. For example, the I-frame in a GoP is required to decode all other P-frames and B-frames in the GoP, and the P-frame is required to decode all the successive P-frames as well as the B-frames encoded with respect to these P-frames. Thus, even if a frame has been received before deadline, it is regarded as *not constructed* if its dependent frames are not reconstructed successfully.

- PSNR (Peak Signal to Noise Ratio). This is a commonly used metric to evaluate the quality of a decoded video. In the video trace file, the PSNR value is computed for each frame based on the difference between the decoded frame and the original frame.

Note that "packet delivery delay", "packet delay jitter", and "packet drop ratio" are network layer metrics, which give us some insight on the perceived video quality, while "percentage of frames reconstructed" and "PSNR" are application layer metrics,



(a) $\alpha = 0$ (b) $\alpha = 1$

Figure 6.4: The Number of Paths Discovered

which directly assess the perceived video quality. In this chapter, we evaluate both categories of metrics.

6.4.2 Number of Paths

We select one router in the network as the receiver and randomly designate n routers in the network as senders. $MinW$ always finds n paths, while the number of paths discovered by $MEDP$ is $min\{n, degree(r)\}$, that is, the network is well-connected so that the number of edge-disjoint paths to the receiver r is bounded by its degree in the topology G_A. For both IPD and PPD, we set the threshold $\alpha = 0, 1$. By setting $\alpha = 0$, we require that each path does not interfere with any other path. When $\alpha = 1$, each link of any path interferes with at most one other path among the multiple paths finally discovered. According to Section 6.2.3, we allow paths to merge at the last hop towards the receiver for both IPD and PPD.

Unlike $MinW$ and $MEDP$, IPD and PPD take wireless interference into consideration when finding multiple paths, and thus have the prospect of providing better video streaming quality. The number of paths discovered by IPD and PPD under

different number of senders is illustrated in Fig.6.4. Each point corresponds to the average of 20 runs. We can observe that PPD is able to discover more paths than IPD under the same constraint. As a result, PPD provides more flexibility for finding appropriate rate allocation to minimize the network congestion than IPD. Especially when $\alpha = 0$, IPD finds only a single path in most cases, while PPD discovers more paths so that we can use multipath streaming to improve the performance of video streaming. Both algorithms tend to find more paths with the increase of available senders.

6.4.3 Number of Sessions

We experiment on networks of different scales (with the same density as the default 60 node topology) without changing the other default settings. In each network, three mesh routers are randomly selected as initial senders. They may be mesh routers connected with a peer that has buffered the video, or a gateway through which a local user can access a peer or media server in the Internet. Assume there are 3 popular movies recently (listed in Table 6.2), and each of the initial senders can provide all of the 3 movies. Assume VoD requests arrive in accordance with Poisson distribution. For each arriving VoD request, the user is connected to a random mesh router in the network, and initiates a VoD request for a movie that is randomly selected from the 3 movies. We use different algorithms to find the routing and rate allocation for each arriving VoD request. If the VoD request can be satisfied and has been established successfully, it becomes a sender, which can provide the movie to subsequent VoD requests. This process continues until there is a VoD request that cannot be satisfied. In this way, we can get the maximum number of concurrent sessions that can be supported in the network.

Fig.6.5 demonstrates the maximum number of concurrent sessions supported by each method under different network scales. For IPD and PPD, α is set to 1. From

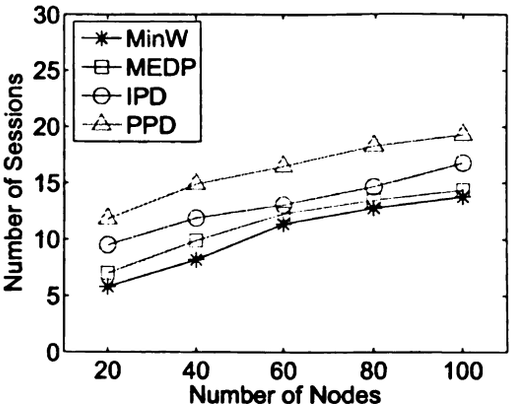


Figure 6.5: Maximum Number of Sessions vs Network Scale (8 channels)

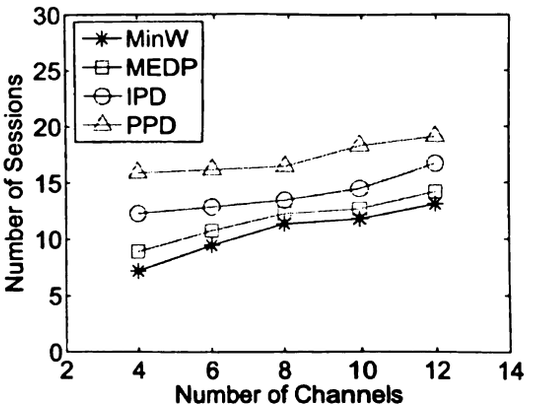


Figure 6.6: Maximum Number of Sessions vs Number of Channels (60 nodes)

the figure, we can observe that $MinW$ performs the worst in supporting multiple sessions. $MEDP$ is slightly better than $MinW$. IPD supports over 10 percent more sessions than $MEDP$ on average, while PPD improves the capacity by over 25 percent compared with $MEDP$. This is because IPD and PPD not only find edge-disjoint paths, but also minimize the interference among the paths. Therefore, the traffic of each session can be well balanced over the network both spatially and over different channels. PPD excels IPD, because PPD finds more paths than IPD under the same constraint, and thus PPD provides more flexibility for rate allocation to minimize network congestion than IPD. Note that the rate allocation algorithm does not mandate every discovered path for a single session to be allocated with some positive rate (explained in Section 6.3). Although PPD finds more paths than IPD, it does not necessarily mean that PPD uses more paths for actual video streaming than IPD. The rate allocation on PPD paths usually has lower congestion than on IPD paths. Therefore, PPD supports more concurrent VoD sessions than IPD.

We repeat our simulations with different numbers of channels. Fig.6.6 illustrates the maximum number of concurrent sessions under different numbers of orthogonal channels for different methods. We can observe that more concurrent VoD sessions can be supported in the network with the increasing number of channels, because the

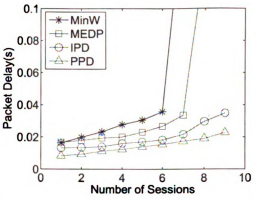


Figure 6.7: Packet Delay over Multiple *VoD* Sessions

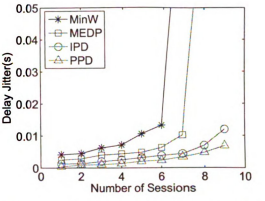


Figure 6.8: Delay Jitter over Multiple *VoD* Sessions

network capacity is increased by using more orthogonal channels.

Note that the maximum number of sessions is not big compared to the 11Mbps bandwidth. This is because there is MAC protocol overhead and the interference is not completely eliminated in the network.

6.4.4 Delay and Jitter

We use the same assumptions of the video sources and the arrival of *VoD* requests as the previous subsection. The video frames are encapsulated into *UDP* packets with the maximum size of 1024 bytes for transmission (similar to [126]). In this scenario, the maximum number of concurrent sessions that can be supported by *MinW*, *MEDP*, *IPD*, *PPD* is 9, 10, 13, 16, respectively. If the network reaches its maximum capacity, additional *VoD* requests will be blocked. We calculate the average packet delay and delay jitter under different numbers of concurrent sessions in the network. The results are shown in Fig.6.7 and Fig.6.8.

In both figures, the horizonal axis shows the number of concurrent *VoD* sessions, and the vertical axis shows the average delay and jitter over all the sessions. As we can see, *IPD* and *PPD* enjoy lower delay and jitter than *MinW* and *MEDP*. For example, when there are 5 concurrent sessions, *IPD* and *PPD* reduce the packet

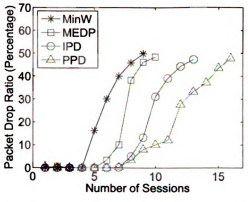


Figure 6.9: Packet Drop Ratio over Multiple VoD Sessions (Playout Deadline = 300ms)

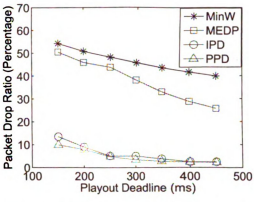


Figure 6.10: Packet Drop Ratio under Different Values of Playout Deadline (8 sessions)

delay by 28 percent and 39 percent, and reduce the delay jitter by 34 percent and 57 percent, compared to $MEDP$. This is because IPD and PPD not only find multiple high-quality paths with minimized interference, but also are better at balancing the traffic in the network both spatially and over different channels, and thereby reducing the network congestion.

6.4.5 Packet Drop Ratio

In this subsection, we first set the playout deadline at the receiver to $300ms$ and calculate the packet drop ratio of VoD sessions. We then vary the playout deadline to evaluate its impact on the video streaming quality. If the receiver streams a video from a gateway, we add a random delay conforming to normal distribution $N(100ms, 20ms^2)$ (the parameters are derived from the packet delays extracted from an experiment where we repeatedly send ping packets from a gateway to a video server) on the path to simulate the delay from the peer in the Internet to the gateway.

Fig.6.9 illustrates the average packet drop ratio over all sessions when there are different numbers of concurrent VoD sessions in the network. We can observe that the drop ratio increases with the increasing number of sessions, because the network

is getting more and more congested. When there are few (less than 5) concurrent session in the network, all methods enjoy very low packet drop ratio. However, when more sessions have been established, IPD and PPD have dramatically lower packet drop ratio than $MinW$ and $MEDP$, because IPD and PPD find better paths for streaming the video. For example, to guarantee an average packet drop ratio less than 20 percent, the maximum number of concurrent sessions that can be supported by $MinW$, $MEDP$, IPD, PPD is 5, 7, 9, 11, respectively.

Fig.6.10 demonstrates the impact of playout deadline on the packet drop ratio. We simulate the case when there are 8 concurrent VoD sessions in the network. The vertical axis shows the average packet drop ratio over all the 8 sessions. We can observe that the drop ratio decreases with longer playout deadline for all the methods, because the playback deadline is relaxed with longer playout deadline. For example, the packet drop ratio of IPD (or PPD) decreases from 14 percent (or 10 percent) to 3 percent (or 2 percent) when the playout deadline increases from 150ms to 450ms. In addition, IPD and PPD leads to much lower packet drop ratio than $MinW$ and $MEDP$, which implies high video quality perceived at receivers. For example, at the playout deadline of 300ms, IPD and PPD reduce the packet drop ratio by around 85 percent, compared to $MEDP$.

6.4.6 Perceived Video Quality

In addition to the network layer metrics that have been analyzed in the previous subsections, we evaluate the application layer metrics of video streaming in this subsection.

Fig.6.11 illustrates the percentage of frames reconstructed under each method when there are different numbers of concurrent VoD sessions in the network. Each point in the figure corresponds to the average of frame reconstruction ratio over all the sessions. The PSNR metric of videos perceived at receivers is calculated and

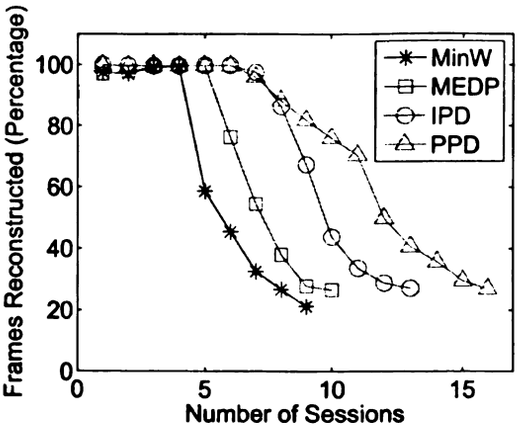


Figure 6.11: Percentage of Frames Reconstructed (Playout Deadline = 300ms)

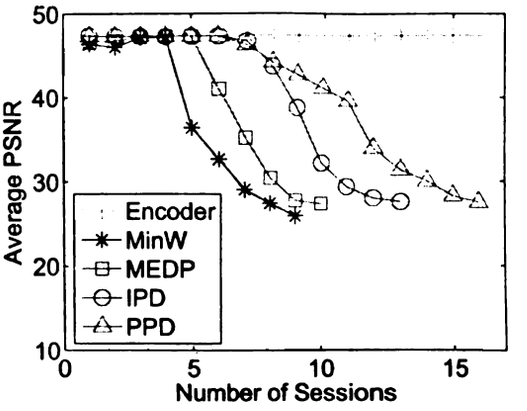


Figure 6.12: Average PSNR over Multiple VoD Sessions (Playout Deadline = 300ms)

demonstrated in Fig.6.12. In the figure, each point corresponds to the average of PSNR over all the sessions. The '+' line plots the PSNR when there is no frame loss in video streaming (optimal value). The PSNR of the 3 movies in Table 6.2 without any frame loss is 47.7 dB, 46.9 dB, and 47.6 dB respectively.

From both figures, we can observer that when there are fewer sessions in the network (less than 5), all the algorithms lead to very high frame reconstruction ratio and high PSNR because of low packet drop ratio. However, when more sessions are established in the network, IPD and PPD obviously excel the other methods, while PPD performs better than IPD. For example, to guarantee an average PSNR no less than 40 dB, the maximum number of concurrent sessions that can be supported by $MinW$, $MEDP$, IPD, PPD is 4, 6, 8, 11, respectively.

Fig.6.13 illustrates the PSNR under different values of playout deadline when there are 8 concurrent VoD sessions in the network. We can observe that the perceived video quality increases with longer playout deadline. For IPD and PPD, when the playout deadline is over 200ms, the improvement of PSNR becomes less obvious. In comparison, with the increase of playout deadline, there is steady improvement of PSNR for $MinW$ and $MEDP$. This is because IPD and PPD have lower delay

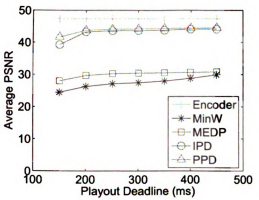


Figure 6.13: Average PSNR under Different Values of Playout Deadline (8 sessions)

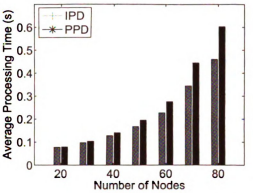


Figure 6.14: Processing Time for Each VoD Request

jitter than $MinW$ and $MEDP$ in network transmission. Therefore, IPD and PPD only need a small playout deadline to reach near-optimal performance.

6.4.7 Processing and Session Setup Overhead

We first evaluate the time needed for computing the routing and rate allocation for each VoD request. The processing overhead includes the discovery of multiple independent paths and the calculation of routing and rate allocation using LP. We apply our algorithms under different network scales. In each scenario, we set the number of senders to 8. We run the algorithms on a laptop computer with 2GHz CPU. The average processing time is shown in Fig.6.14. We can observe that PPD has a little longer processing time than IPD. However, PPD brings better performance than IPD with regard to the average session quality and the network's capacity based on the analysis in previous subsections. Under the network scale of 60 nodes, both PPD and IPD are fast enough (less than $300ms$) to calculate the routing and rate allocation for each VoD request.

We also calculate the time needed for setting up the session once the routing and rate allocation have been determined for a VoD request. This includes reserving

the bandwidth on the paths and letting the routers along the paths update the link status to the other routers. The simulation results in a 60 nodes network show that the session setup delay is less than $400ms$. Therefore, the total delay for processing and session setup is less than $700ms$ for each VoD request.

6.5 Summary

The objective of this chapter is to improve the performance of multi-source VoD applications in multi-channel multi-radio wireless mesh networks. We first proposed two heuristic multipath discovery algorithms, IPD and PPD, to find multiple independent paths from senders to the receiver for each VoD request. The proposed algorithms consider wireless interference in the multipath discovery, so it is able to balance the video streaming traffic both spatially and on different channels in the network. Based on the multipath discovery algorithms, we then proposed a joint routing and rate allocation algorithm to find the routes and rate allocation with the goal of minimizing the network congestion. The algorithm not only optimizes the performance of existing VoD sessions in the network, but also improves the network's capability of satisfying more subsequent VoD requests. We performed simulations in $NS2$ using real video traces, and evaluated the performance of our algorithms using both network layer metrics and application layer metrics for video streaming. Simulation results have shown that IPD and PPD not only increase the maximum number of concurrent VoD sessions that can be supported in the network, but also improve the video streaming quality of each session, compared to previous work. Moreover, PPD achieves better video streaming performance than IPD, because it is able to discover more paths than IPD under the same constraint, and therefore provides more flexibility in finding rate allocation with minimized congestion.

CHAPTER 7

Static-Node Assisted Routing in Vehicular Ad-Hoc Networks

A vehicular ad-hoc network can be regarded as a special type of mobile ad hoc network. Several routing protocols have been specially designed for vehicular networks. MDDV [111] and VADD [123] are two multi-hop routing protocols in vehicular networks, which combine geographic forwarding, opportunistic forwarding, and trajectory based forwarding to deliver data from a mobile vehicle to a static position beside the road. They abstract each road as a link where the packet delivery delay depends on the vehicle density on the road. Therefore, a packet will be delivered along the shortest delay trajectory to the destination. However, as a vehicular network consists of highly mobile nodes, it is possible that when a packet reaches an intersection it cannot be delivered along the shortest delay trajectory because there is no vehicle within wireless communication range in the next road along the trajectory to relay the packet at that moment. VADD copes with this problem by selecting the best currently available path, where an available path means there are vehicles within communication range on that path to continue delivering the packet.

There are two problems in the current routing protocols for vehicular networks.

(1) The performance of VADD is quite sensitive to the vehicle density. Under high vehicle density, there is high probability that the next path along the shortest-delay trajectory towards the destination is available when the packet reaches an intersection. However, when the vehicle density is median or low, the second best, third best, or even the worst path may be taken at an intersection because they are the only available paths at the moment. This makes the actual packet delivery route deviate far from the optimal one, leading to dramatic increase in packet delivery delay. (2) The real shortest-delay trajectory may not be taken under inaccurate delay estimation of each road. Both MDDV and VADD estimate the packet delivery delay along each road based on some offline statistical parameters, such as the average vehicle velocity and vehicle density on the road at different times of a day. Nevertheless, as these parameters vary with time, the statistical result is not an accurate estimation of the current packet delivery delay along each road.

In this chapter, we propose SADV, a Static-node assisted Adaptive data Dissemination protocol for Vehicular networks. Different from MDDV and VADD, SADV focuses on data delivery in large scale and dynamic VANET under low vehicle densities, where VADD experiences dramatic performance degradation in packet delivery delay, and MDDV even renders poor reliability. The contributions of SADV are in the following aspects.

- We propose to deploy static nodes at intersections, which enables packets to be stored at an intersection until the optimal path (the path with the minimum expected data delivery delay to the destination) is available. Our simulation shows that the proposed static-node assisted routing protocol can dramatically improve the packet delivery performance, compared to previous work.

- To better estimate the delay of forwarding packets along each road, the static nodes are designed to be able to measure the packet forwarding delay and propagate the link delay information. Therefore, the routing decision in each static

node can adapt to the changing vehicle densities. We analyze the converging speed of link delay update by simulations.

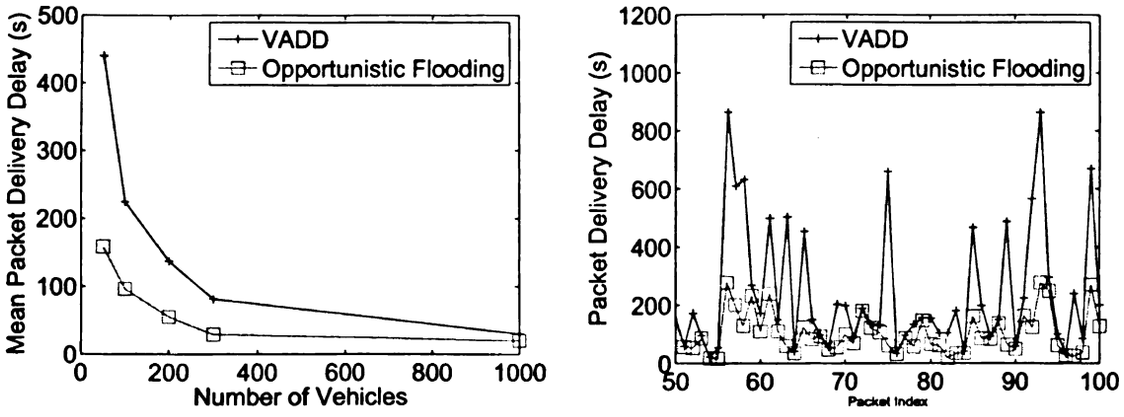
- We also propose a multi-path routing algorithm, which can further decrease the packet delivery delay without exponential increase of protocol overhead.
- Furthermore, we study intelligent partial deployment strategies of static nodes when it is not possible to deploy a static node at each intersection.

The rest of the chapter is organized as follows. The motivation of SADV is introduced in Section 7.1. The detailed description of the routing protocol is in Section 7.2. SADV under partial deployment of static nodes is discussed in Section 7.3. In Section 7.4, we evaluate our proposed approach by comparing it with VADD. We summarize this chapter in Section 7.5.

7.1 Data Dissemination under Different vehicle Densities

To evaluate the performance of VADD with regard to packet delivery delay, we use the delay of epidemic flooding [105] as a baseline. In epidemic flooding, two vehicles exchange the packets that they do not have in common whenever they can communicate with each other. Assume there is no packet loss due to transmission collision, epidemic flooding is the quickest way to deliver a packet to its destination. Therefore, we use this value as the lower bound in our analysis. In the following of this chapter, we refer to this method as "flooding".

Fig.7.1 illustrates our experiment to evaluate the performance of VADD under different vehicle densities. We extract a regional road map from TIGER [4]. The simulation is performed on this road topology with different numbers of vehicles (The detailed experiment setting is explained in Section 7.4). As shown in Fig.7.1(a),



(a) Mean Packet Delivery Delay

(b) Packet Delivery Delay (100 Vehicles)

Figure 7.1: Packet Delivery Delay of VADD

when the vehicle density is high, the mean packet delivery delay of VADD is close to that of Flooding. However, the gap increases with the decrease of vehicle density. Moreover, VADD becomes unstable under low vehicle densities. Fig.7.1(b) shows the comparison of the delivery delay of individual packets between VADD and Flooding when 100 vehicles are simulated in the area. We can observe that the packet delivery delay of some packets is much larger than that of Flooding while the delay of some other packets is close or even equal to the optimal value.

There are two reasons for this performance degradation. (1) VADD chooses the best currently available path at each intersection. However, when the vehicle density is low, the optimal path may not always be available at the moment. Thus, VADD has to deliver packets via detoured paths. In the worst case, the packet may go through a much longer path as indicated by the peaks in Fig.7.1(b). (2) The estimation of packet forwarding delay through each road is based on some offline statistical data such as the average vehicle density, because it is expensive to have each vehicle get up-to-date vehicle densities from some infrastructures. As the vehicle density on each road may vary with time, which greatly influences the packet forwarding delay, the shortest-delay path calculated based on the statistical data may not reflect the

real optimal one. Due to these reasons, we propose SADV in this chapter, which can improve the performance of data dissemination under low vehicle densities in VANET.

7.2 SADV Design

As the shortest delay path is not always available at the moment when a packet reaches an intersection, we can deploy a static node at each intersection to assist packet delivery. The static node can store the packet for some time until the shortest delay path becomes available. As illustrated in Fig.7.2, a packet is sent from A to a remote location. Once the packet is relayed from A to B, B needs to determine the next vehicle to forward. Assume the shortest delay path to deliver the packet is northward. However, there is no vehicle within communication range of B that can deliver the packet along the direction. Thus B relays the packet to the static node. The static node stores the packet for a while, and forwards it to C when it passes the intersection and goes towards the northward road. From the figure, we can see that without the help of the static node, the packet will be carried by B to the eastward road if B does not meet C at the intersection, which may lead to a much longer packet delivery path.

We assume that each vehicle knows its location through the GPS service, which is already available in most new cars and will be common in the future. Each vehicle or static node is equipped with a radio capable of short range wireless communication (100m-250m). All vehicles and static nodes broadcast beacons messages periodically, so that they can know their neighboring vehicles and static nodes. In addition, each static node knows its own position and has a digital street map, based on which the packet forwarding trajectory is determined. In this section, we focus on the critical issue of how to reduce the packet delivery delay by the assistance of static nodes.

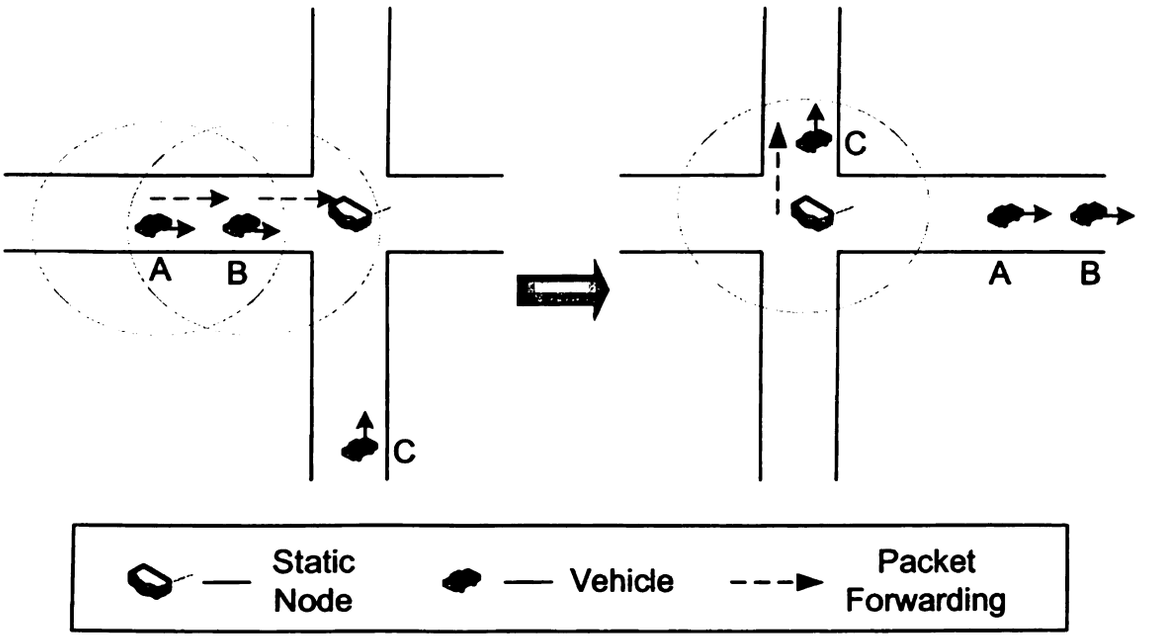


Figure 7.2: Static Node Assisted Routing in VANET

Efficient routing algorithms need to be designed for both vehicles and static nodes, while cares must be taken to alleviate the loads in static nodes. In the following, we present the design of SADV, a Static-node assisted Adaptive data Dissemination protocol for Vehicular networks. It consists of three modules: SNAR (Static Node Assisted Routing), LDU (Link Delay Update) and MPDD (Multi-Path Data Dissemination).

7.2.1 Static Node Assisted Routing

1. System Model

Given the road map of the region, we can abstract it as a directed graph $G(V, E)$, where V is the set of intersections and E is the set of directed roads. Assume each intersection v_i is deployed with a static node s_i. Let s_i and s_j be the static nodes at two adjacent intersections v_i and v_j, then the expected delay of delivering a packet from s_i to s_j through the vehicles on road $v_i v_j$, denoted as $d(s_i s_j)$, can be calculated

as:

$$d(s_i s_j) = w(s_i s_j) + t(v_i v_j) \tag{7.1}$$

where $w(s_i s_j)$ denotes the expected waiting time of a packet at s_i for vehicles coming into communication range to deliver the packet toward s_j along road $v_i v_j$, and $t(v_i v_j)$ denotes the expected time taken to deliver a packet through road $v_i v_j$ given that vehicles are currently available for transmitting the packet at s_i along $v_i v_j$.

Suppose vehicles enter road $v_i v_j$ from v_i with an average rate of λ_{ij}, then $\lambda_{ij} = v_{ij} \times \rho_{ij}$, where v_{ij} and ρ_{ij} represent the average vehicle velocity and average vehicle density on the road, respectively. Therefore, the expected waiting time can be calculated as:

$$w(s_i s_j) = \frac{1}{\lambda_{ij}} = \frac{1}{v_{ij} \times \rho_{ij}} \tag{7.2}$$

On the other hand, $t(v_i v_j)$ is dependent on the vehicle density, the vehicle velocity, the length of road, and the maximum wireless transmission range. We use the following formula in [123] to calculate this value, where l_{ij} denotes the length of road $v_i v_j$, and R represents the maximum wireless transmission range.

$$t(v_i v_j) = \begin{cases} \alpha \cdot l_{ij} & \text{if } \frac{1}{\rho_{ij}} \leq R \\ \frac{l_{ij}}{v_{ij}} - \beta \cdot \rho_{ij} & \text{if } \frac{1}{\rho_{ij}} > R \end{cases} \tag{7.3}$$

If the average distance between vehicles is smaller than R, the packet is mainly forwarded by wireless transmission (α can be regarded as the reciprocal of wireless transmission speed), which is much faster than the carry of vehicles. Otherwise, the packet is mainly relayed by the carry of vehicles. In the equation, $\beta \cdot \rho_{ij}$ is used as a correction factor, because it is still possible to have occasion wireless transmission between some vehicles on the road in this case. As the movement of vehicles is directional, the packet forwarding delay from s_i to s_j, denoted by $d(s_i s_j)$, is often different from the forwarding delay in the reverse direction, denoted by $d(s_j s_i)$, because the vehicle densities in the two directions are different.

Assume each static node has a digital map of the region, it can then abstract it as a directed graph G and generate a delay matrix d on G, where $d_{ij} = d(s_i s_j)$. The static-node assisted routing aims at reducing the expected delivery delay of data packets. We define the optimal path as the path with the minimum expected data delivery delay based on delay matrix d, then we try to ensure that the data packets are delivered along the optimal paths.

2. SNAR overview

In SNAR, the packet delivery service is accomplished by the relay of both intermediate vehicles and static nodes at intersections. It is composed of three modes: *vehicle in-road mode*, *vehicle intersection mode*, and *static-node mode*.

- When a data packet is in a vehicle that is not within wireless transmission of the static node, the SNAR stack on the vehicle works in *vehicle in-road mode*. In this mode, the packet will be relayed by the carry of vehicles or the wireless transmission to other vehicles towards the next intersection (or static node).

- When a data packet is in a vehicle that reaches within the wireless communication range of the static node, the SNAR stack on the vehicle works in *vehicle intersection mode*. In this mode, the packet may be delivered to another vehicle in the intersection area, or delivered to the static node, or still carried by the vehicle instead of being delivered.

- When a packet is in a static node, the SNAR stack on the static node works in *static-node mode*. It will forward the packet to an appropriate vehicle if available.

The basic operation of SNAR is illustrated in Fig.7.3.

We use the assistance of static nodes to guarantee that the data packet is delivered along the optimal path. Once a packet reaches the static node (it may be in the static

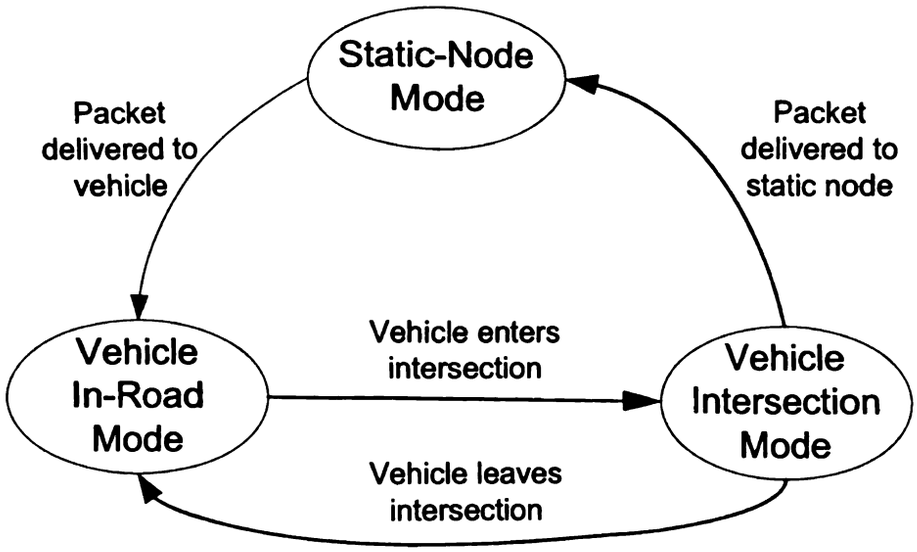


Figure 7.3: Operation Modes of SNAR

node or a vehicle within R of the static node), let s_t be the nearest static node of the packet's destination, then s_i can calculate the optimal path from s_i to s_t based on delay matrix d. Thus, SNAR should deliver the packet towards the next static node along the optimal path. In this way, the packet can be delivered along the minimum expected delay path.

3. Data Forwarding in Vehicles

SNAR operates in two modes, that is, in-road mode and intersection mode, when a packet is in a vehicle.

Upon reaching the intersection (or the vehicle is in radio communication range of the static node in the intersection), the SNAR stack on the vehicle works in intersection mode. Assume the vehicle has packets with destination dst. Denote the static node at the intersection by s_i. The vehicle first sends a routing query to s_i for destination dst. Node s_i calculates the optimal next intersection, denoted by s_j, that the packet should be delivered to, and replies to the vehicle. (As the calculation of the optimal path in static nodes is by shortest path algorithm, it only takes several

Algorithm 10 Packet Delivery for Vehicle Intersection Mode

1: Denote the vehicle by c, and the static node by s_i.
2: Assume c has packets destined for dst.
3: Query s_i for the next intersection for dst, denoted by s_j.
4: **while** c has not passed the intersection **do**
5: Let $N(c)$ denote the neighbors of c that are going towards direction s_is_j ($c \in N(c)$). These can be vehicles already in road s_is_j, or vehicles that will turn into s_is_j after passing s_i.
6: **if** $N(c)$ is not empty **then**
7: For $c_k \in N(c)$, calculate $d_k = (-1)^p \cdot distance(c_k, s_i)$, where $p = 0$ if c_k is in road s_is_j now, or $p = 1$ otherwise (c_k has not entered the road yet).
8: Let $k^* = argmin(\{d_k\})$.
9: **if** $c_{k^*} \neq c$ **then**
10: Deliver the packets to c_{k^*}.
11: **return**
12: **end if**
13: **end if**
14: **end while**
15: **if** c is not going towards s_is_j **then**
16: Deliver the packets to s_i.
17: **end if**

milliseconds to finish the query.) The vehicle then determines the next vehicle to deliver the packet to. It will forward the packet to the vehicle that is both moving towards s_j and closest to s_j. If such vehicle exists, the packet will be delivered to the next hop. A special case is that the vehicle itself is the best vehicle to further deliver the packet, so the vehicle will continue to carry the packet. If there are no such vehicles, when the vehicle starts to leave the intersection, it will deliver the packet to s_i, so that the static node can deliver the packet along the best direction when there are vehicles available. The algorithm is illustrated in Algorithm 10.

When a packet is carried by a vehicle in a road, which has not entered the communication range of static nodes yet, the SNAR stack on the vehicle operates in in-road mode. Assume the next adjacent intersection to deliver the packet is s_i, we use geographic forwarding to deliver the packet greedily to s_i, that is, if the vehicle

finds

to th

4. Da

Whe

It wil

carry

static

perio

scan

path

vehi

it wi

upd

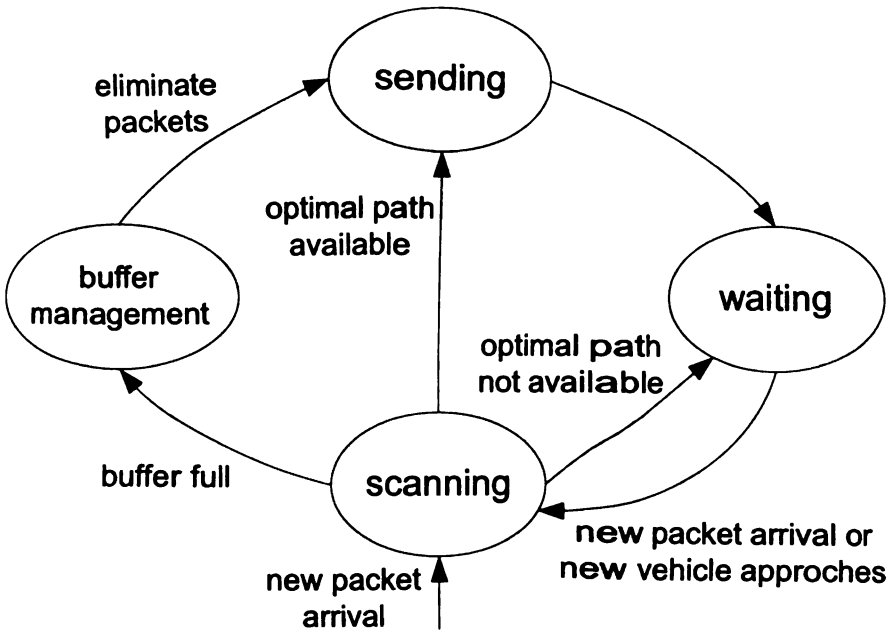


Figure 7.4: State Transition Diagram of the Intersection Mode

finds another neighboring vehicle, which is more close to s_i, it will deliver the packet to that vehicle.

4. Data Forwarding in Static Nodes

When a packet is in a static node, the SNAR stack on it works in static-node mode. It will deliver the packet along the optimal path when there are vehicles available to carry the packet to that direction. Fig.7.4 shows the state transition diagram of the static-node mode. The static node knows its neighboring vehicles by listening to the periodic beacon messages from vehicles. Upon arrival of new packets, SNAR enters *scanning* state. It looks for the vehicles that can carry the packet along the optimal path in its neighbor set. If there are such vehicles, it will deliver the packet to the vehicle that is farthest in the optimal direction (similar to Algorithm 10). Otherwise, it will enter *waiting* state, and return to scanning state when the neighbor set gets updated because of newly entered vehicles.

As packets may be stored in the static node before being delivered further, buffer management is a critical issue when the buffer turns out to be full in the static node. In this case, some packets must be eliminated from the buffer. Instead of being directly dropped, these packets can be delivered immediately through the best currently available paths, without decreasing the packet delivery reliability.

Several buffer elimination strategies can be used.

- *FIFO (First In First Out)*, where the packets that stay the longest in the buffer are eliminated first. However, this is not a good strategy because these packets may have higher probability of being forwarded along the optimal path after waiting for quite a long time in the static node.

- *FILO (First In Last Out)*, where the most recently arrived packets are eliminated first. However, as these packets have just arrived, the best or even the second best path is possibly not available yet. As a result, the packets may be routed along some much longer paths.

- *Least Delay Increase*, which aims at reducing the increase on the overall packet delivery delay caused by sending packets along sub-optimal paths. The basic idea is as below. Suppose the static node is at intersection v_i. In this node, we define a priority vector $[p_1, p_2, ..., p_m]$ for each packet, where m is the number of adjacent roads of v_i, and p_j is the ranking of the optimality of the corresponding road. For example, if v_i has 4 adjacent roads, $[2, 1, 3, 4]$ means the first adjacent road corresponds to the second best path, while the second adjacent road corresponds to the best path, and so on. The static node will scan for currently available roads, thus the rank of the currently best path can be determined for each packet, which we call *instant rank*. As for the example above, if the first and fourth adjacent roads are available, then the first road corresponds to the currently best path of delivering the packet, and thus the instant rank is 2. In

our strategy, the packets with the highest instant rank will be eliminated from the buffer first. Instead of being directly dropped, these packets will be forwarded along the best currently available paths. Intuitively, this strategy tries to forward some packets along sub-optimal paths in the hope that they will not differ significantly from the optimal paths. Therefore, this strategy is favorable to preserve low packet delivery delay in VANET.

7.2.2 Link Delay Update

In the previous subsection, the routing decision is made on a delay matrix, where the delay of each link is estimated based on the statistical vehicle density over a long period of time in each road. However, the estimation is often inaccurate because of the variability of vehicle density with time. In this subsection, we introduce LDU module, which measures the delay of each link in real time and propagates the up-to-date estimation among static nodes.

Let s_i and s_j be two adjacent static nodes. We can measure the delay of $s_i s_j$ by piggybacking some fields in the data packet that is forwarded from s_i to s_j. To measure the delay, we insert a single field *stime* into the packet head. When s_i receives the packet for further delivery, it immediately records the current time in *stime*. Then when the packet reaches s_j, s_j can calculate the measured packet forwarding delay:

$$d'(s_i s_j) = etime - stime$$

where *etime* is the current time. Therefore, each static node s_j is able to obtain the measured delay of all its incoming links $\{s_i s_j\}$.

Let the delay matrix maintained by each s_i be d^i. Initially, the delay of each link in the matrix is calculated by Equation (1) based on statistical data. Let $\overline{d'}(s_j s_i)$ be the mean of the measured delay $d'(s_j s_i)$ over time interval I. Each s_i can obtain $\overline{d'}(s_j s_i)$ for all of its incoming links. In SADV, each static node exchanges with other nodes

the mean measured delay it has measured, so that each node gets a more complete up-to-date delay matrix.

The propagation of the mean measured delay observed in each static node is in the form of *delay update message*, which consists of records in the following form:

$$< src\_id,\ dst\_id,\ delay,\ seq,\ expire >$$

where *src_id* and *dst_id* are the IDs of the starting and ending static node, *delay* is the mean measured packet forwarding delay from *src_id* to *dst_id*, *seq* is the sequence number indicating the freshness of the message, and *expire* denotes the time when the information in the message becomes invalid.

Once each static node s_i obtains $\overline{d'}(s_j s_i)$, it encapsulates them into the *delay update message*, and informs other nodes by broadcast. The broadcast is realized by wireless transmission or the carry and forward of vehicles from one static node to its adjacent nodes. The process is similar with *Link-State Broadcast* [58]. To prevent flooding the network, we limit LDU in the local area. For example, we set the TTL of each LDU packet to K, which is decreased by 1 in each static node. The optimal path will be recalculated at each static node, so once the packet gets near the destination, it will get more accurate optimal paths. To further reduce overhead, each static node only broadcasts the freshest delay measurement for the same link, which is identified by *seq*, and each message is broadcasted only once at each node. Upon receiving a delay update message, each static node modifies its delay matrix accordingly. Suppose s_k receives the delay measurement of the link $s_m s_n$, then d^k will be modified by the following formula.

$$d^k(s_m s_n) = k_1 \cdot d^k(s_m s_n) + k_2 \cdot \overline{d'}(s_m s_n)$$

where k_1 and k_2 are two coefficients satisfying $k_1 + k_2 = 1$.

7.2.3 Multi-Path Data Dissemination

In classical studies, multi-path routing has often been used to improve data delivery reliability or achieve load balance in network communications. In this chapter, we are interested in decreasing the packet delivery delay by routing packets through multiple paths.

In the VANET, if the packet forwarding delay between each two adjacent static nodes cannot be estimated correctly, the shortest-delay path computed based on the estimated delay matrix may not be the real optimal one. As the packet forwarding delay through different roads may vary dramatically, depending on the vehicle distribution on the roads, the different paths that a packet goes through may differ greatly in the total delivery delay. This motivates us to utilize multi-path routing, when the packet load in the VANET is not high, to decrease the packet delivery delay by trying to hit a faster path than the single-path routing.

Packets are delivered through multiple paths only at intersections. Therefore, multi-path routing can be realized by only some additional support in static nodes, without any modifications of the protocol stack in vehicles. Assume a packet is in s_i at present. Let $N(s_i)$ be the set of static nodes that are adjacent to s_i. Node s_i selects a subset of $N(s_i)$, denoted by $N_s(s_i)$, and then sends a copy of the packet to each static node in $N_s(s_i)$. In this way, the packet will be delivered along multiple paths. In order to reduce the overhead caused by multi-path routing, we let each intermediate static node s_k remember the packets it has sent out for some time T_m. If the same packet arrives at s_k from other paths later, it simply ignores this packet.

We use the following strategy to choose $N_s(s_i)$ out of $N(s_i)$. Let s_i compute the priority vector $[p_1, p_2, ..., p_m]$ for the packet. Then s_i will send the packet through the roads corresponding to the best and second best paths. For example, if the priority vector is $\boldsymbol{p} = [2, 3, 1, 4]$, then the packet will be forwarded to both the first and third adjacent static nodes. This strategy applies when the estimation of the link delay

is not accurate, so that it will increase the chance of hitting a better or even the real optimal path. As most paths converge in the end and a static node will ignore the packet if it has relayed it before, the multi-path packet delivery does not incur exponential explosion of packet delivery overhead.

7.3 Partial Deployment of Static Nodes

In the previous section, we assume that each intersection is equipped with a static node. However, when it is impossible to put a node at each intersection, the node deployment strategy becomes important, which greatly influences the average packet delivery performance in the VANET.

In this subsection, we give some heuristic principles for node deployment. We will analyze these node deployment strategies by simulations in the next section.

- The static nodes should be deployed uniformly. This is quite intuitive. As the vehicle density varies with time and space, if we give more preference to a specific area, then the packet delivery performance in other areas will be poor.

- The intersections with more connected roads should preferably be equipped with static nodes. The reason is that if the degree of the intersection is high, a packet has more choices of routes to be forwarded along when it reaches the intersection. If the nearby vehicle density happens to be low and the intersection is without any static node, the packet will probably miss several better paths. However, if the intersection is equipped with a static node, the packet can be stored in the node to wait for a much better choice. In other words, the deployment of static nodes in these intersections can lead to more dramatic improvement in packet delivery performance than those lower-degree intersections.

- The intersections along high-speed roads should preferably be deployed with static nodes. As the average speed is higher and there are usually more vehi-

cles in the main roads because vehicles tend to take these faster roads to get to their destinations, the packet delivery delay through these roads is usually lower than the other small roads. Thus, more packets tend to be delivered to destinations via these high-speed roads. However, if a packet is delivered to an intersection of the main road from some small roads, and there are no vehicles within communication range to continue forwarding the packet along the main road at the moment, the packet will miss this best delivery path and be routed through other small roads, which increases the packet delivery delay substantially. Therefore, deploying static nodes at these intersections may have a dramatic effect in improving the packet delivery performance.

7.4 Performance Evaluation

We perform the simulation on a real street map, which we extract from TIGER [4]. The Tiger database contains detailed information of each road, such as the positions of the two ends of each road and the type of each road. We extract a regional urban area from the Tiger database with the range of $4000m \times 5000m$, which contains 70 intersections. The speed limit of each road is determined based on the road type. In the figure, there are two high-speed roads (painted in bold) with speed limits over $50mph$, and the other roads are of local speed ranging from $25mph$ to $45mph$. As the Tiger database does not identify one-way streets, we regard all roads in the map as two-way.

We implement the simulation program in MATLAB. We evaluate our protocol under different numbers of vehicles in the area. The mobility model for each vehicle is based on [92]. When the simulation starts, each vehicle determines a random destination, and then selects a quickest path or a shortest path with equal probability to the destination. Upon arriving at its destination following the pre-determined path,

Table 7.1: Simulation Setup

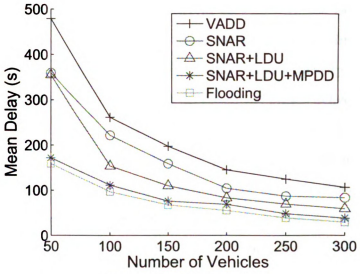
Simulation Area	$4000m \times 5000m$
Number of Intersections	70
Number of Vehicles	50, 100, 150, 200, 250, 300
Vehicle Velocity	$25 \sim 65$ miles per hour
Radio Communication Range	200m
Data Packet Length	1024 bytes
Vehicle Beacon Interval	0.5 sec
LDU parameters	Interval=10 minutes, TTL=10

each vehicle reselects a random destination and moves on again. In our experiments, we assume that each vehicle or static node has a wireless communication range of 200 meters, and broadcasts a beacon message every 0.5 second so that each vehicle or static node can get an updated list of neighbors. For data packet generation, each packet is generated by a random vehicle with a random position in the map as destination. The data packet size is set to 1024 bytes. The parameters are listed in Table 7.1.

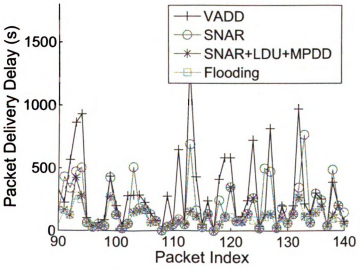
7.4.1 Packet Delivery Delay

We randomly select vehicles to generate packets with random destinations in the map. We control the total packet generation rate in the map at 10 packets per second. We use different data dissemination protocols to deliver the packets, and finally compute the mean packet delivery delay of all the packets in order to compare different protocols. In this subsection, we assume that each static node has enough buffer such that no packet elimination occurs. We will evaluate SADV with limited buffer capacity of static nodes later. The total simulation time is set to one hour. The simulation is repeated under different vehicle densities, and the results are shown in Fig.7.5.

As shown in Fig.7.5(a), the bottom curve corresponds to the performance of Epidemic Flooding (we note it as Flooding in this chapter), which can be regarded as the lower bound of packet delivery delay in vehicular networks. VADD suffers from high



(a) Mean Packet Delivery Delay



(b) Delay for an Individual Packet

Figure 7.5: Packet Delivery Delay of SADV

packet delivery delay, especially under low vehicle densities. When 50 vehicles are simulated in the area, the mean packet delivery delay of VADD is more than twice of the lower bound. This is because VADD may route packets through detoured paths instead of the optimal paths. SNAR delivers packets with the assistance of static nodes at intersections. If LDU is not used, the packet forwarding delay through each link can only be estimated by the statistical vehicle density, which is similar with VADD. We can observe that SNAR decreases the packet delivery delay to some extent, which indicates that the static nodes can help improve the packet delivery performance. If we use LDU to inform the static nodes of the up-to-date packet forwarding delay through each link, the performance can be further improved. One exception is that when the traffic density is very low, such as below 100 vehicles, LDU does not benefit the packet delivery performance that much, because the link delay update message may not be propagated to the nearby static nodes in time. Furthermore, when MPDD is used, the performance can be further improved, which is close to that of Flooding.

Fig.7.5(b) shows the delivery delay of individual packets by using different routing protocols. In this experiment, the traffic density is fixed at 100 vehicles. As can be seen from the figure, VADD sometimes experiences very large delay. In contrast, SNAR is much better at avoiding delivering packets through very long paths. However, we can observe that the delay of SNAR is higher than VADD for a few packets. This is because SNAR did not compute the real optimal paths for these packets due to the inaccurate link delay estimations. However, if LDU is used with SNAR, the packet delivery delay is lower than VADD for each packet. Moreover, if LDU and MPDD are both used, the packet delivery delay of most packets is close or equal to the lower bound.

Number of Data Transmissions

30

20

10

7.4

Fig

the

nu

VA

tra

Flo

the

wh

per

de

a r

wit

exa

tra

be

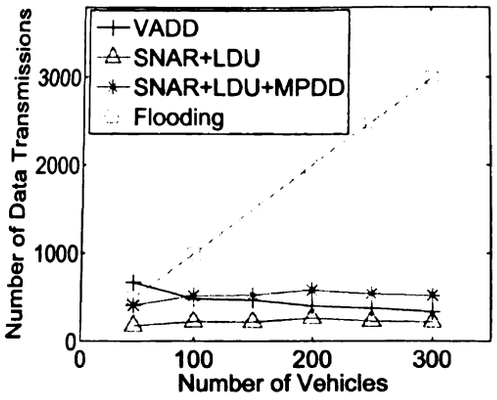


Figure 7.6: Data Packet Overhead

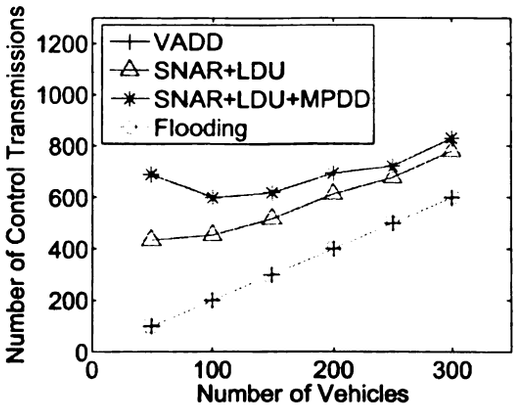


Figure 7.7: Control Packet Overhead

7.4.2 Data Transmission and Protocol Overhead

Fig.7.6 illustrates the data transmission overhead of different protocols. We keep the packet generation rate to 10 packets per second. We then calculate the total number of data transmissions over the whole vehicular network in each second. For VADD, each data transmission only occurs between vehicles. For SADV, the data transmission is either between vehicles or between a vehicle and a static node. For Flooding, every two vehicles exchange all messages they do not have in common when they encounter each other.

From the figure, we can observe that VADD has higher data transmission overhead when there are less vehicles. For example, the overhead decreases by around 50 percent when the number of vehicles increases from 50 to 300. Under lower vehicle densities, a packet possibly cannot be delivered along optimal paths in VADD. As a result, a packet may be delivered to the destination via more vehicles. Compared with VADD, SNAR+LDU has dramatically lower data transmission overhead. For example, when there are 200 vehicles, SNAR+LDU decreases the number of data transmissions by around 40 percent. The reason is that a packet is guaranteed to be delivered along the optimal path with the assistance of static node. The data

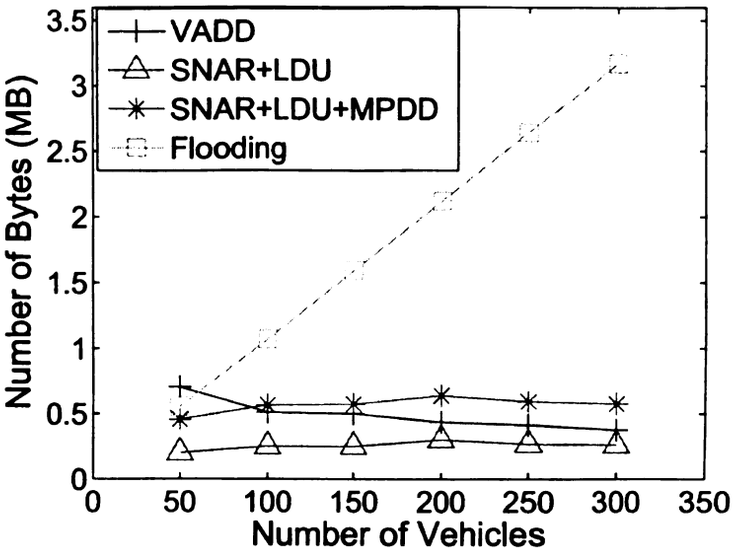


Figure 7.8: Total Overhead

transmission overhead increases by using SNAR+LDU+MPDD, because data packets are delivered through multiple paths. However, it will not result in sharp increase as compared to Flooding, because the multiple paths will converge to the destination.

Fig.7.7 presents the control packet transmission overhead of different protocols. The control overhead of VADD only contains the HELLO packets broadcasted periodically (every 0.5s) by each vehicle. For SADV, more control packets are needed in two facets. (1) Once a vehicle gets into the wireless communication range of the static node, it will query the static node for the routing of the packets it is holding. (2) Each static node will broadcast its link delay measurement to other static nodes in the local area periodically. For Flooding, control overhead only includes HELLO packets (every 0.5s), which is the same as VADD.

Although SNAR+LDU causes more control overhead, it decreases both the data transmission overhead and the data delivery delay dramatically, compared to VADD. As data packets (1024 bytes per packet) are much longer than control packets ($20 \sim 80$ bytes per packet), SNAR+LDU actually reduces the overall overhead. The total

Ratio of Matching Optimal Paths

Fi

nu

7.4

As

per

del

tha

fast

we

veh

loca

ima

path

whi

pro

tim

W

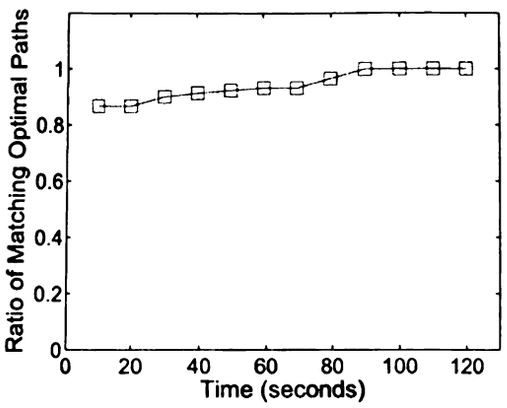


Figure 7.9: Convergence of LDU (optimal path)

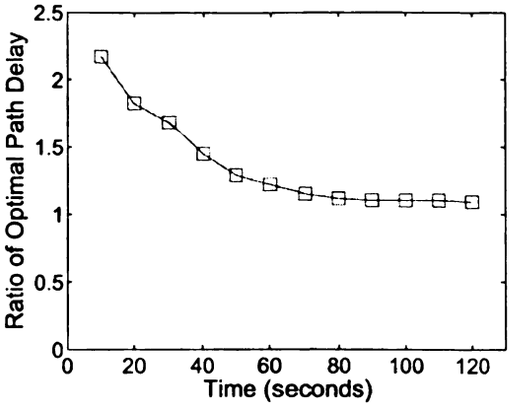


Figure 7.10: Convergence of LDU (optimal paths delays)

number of bytes for both data and control packet transmission is shown in Fig.7.8.

7.4.3 Convergence of LDU

As discussed in Section 7.2.2, we let static nodes propagate the measured link delays periodically (every 10 minutes) so that each static node can get more accurate link delay estimations. To prevent flooding, we set TTL to 10 in each update message so that link delays are updated in the local area. In this subsection, we analyze how fast the link delay matrix can get updated in each static node. In the experiment, we first set 100 vehicles in the road map, and then add 100 more vehicles. Once the vehicle density changes, we compute paths based on two link delay matrixes. (1) The local delay matrix in a static node, which is updated by LDU with time. (2) An imaginary global delay matrix, where each link delay is measured in real-time. The paths computed based on the global delay matrix should be the real optimal path, which we will use as the benchmark to compare with the paths calculated by our proposed method. As there is delay in the LDU, the local delay matrix needs some time for convergence.

We first compare the optimal paths computed from the local delay matrix and

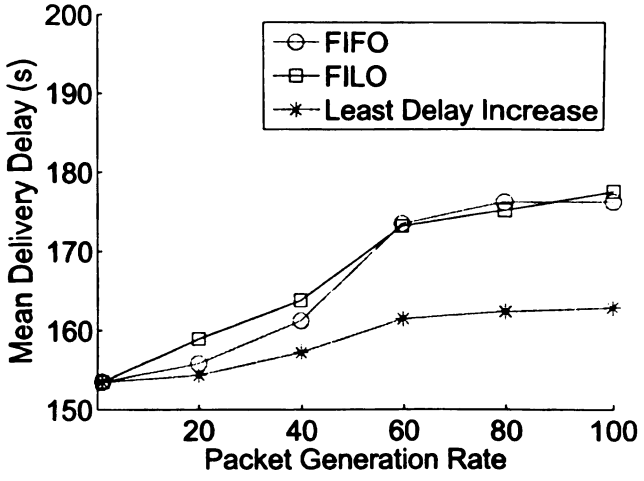


Figure 7.11: Analysis of Buffer Management Strategies

the global delay matrix. There is a matching optimal path if the paths computed from both matrixes are the same. We test with 100 random source-destination pairs in each time. Fig.7.9 shows the percentage of matching paths. We can observe that the percentage of matching paths increases with time, because the local matrix is becoming more accurate when the delays of more links get updated by LDU. We also calculate the ratio of the delay of the optimal path computed from the local matrix over that of the global matrix, and the results are illustrated in Fig.7.10. The ratio of 1 means the local matrix can give paths with the same estimated delay as the global matrix. We can observe that the ratio decreases to almost 1 after 80 seconds, which indicates that the LDU needs 80 seconds to converge. This is a reasonable amount of time, because the vehicle traffic is usually stable in hours.

7.4.4 Buffer Management in Static Nodes

We limit the buffer capacity of each static node, and evaluate SADV (SNAR+LDU) with different packet elimination strategies under different vehicle densities. We set the buffer capacity to 55 packets for each static node, and the number of vehicles to 100. We vary the total packet generation rate in the network, and compare the mean

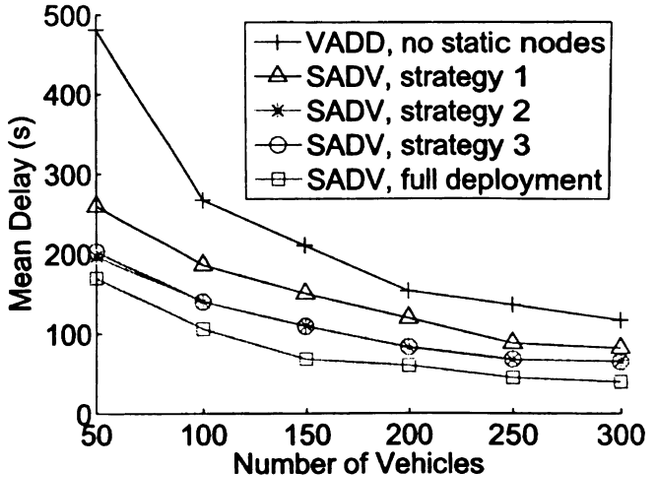


Figure 7.12: Analysis of Node Deployment Strategies

packet delivery delay corresponding to different packet elimination strategies. The simulation results are shown in Fig.7.11. When the packet generation rate is low, the buffer is rarely overflowed. Thus, the performance is similar for different elimination strategies. However, when the packet generation rate is increased, the Least Delay Increase strategy apparently outperforms the other two strategies.

7.4.5 SADV under Partial Deployment of Static Nodes

In our simulation setting, there are a total of 70 intersections in the road map. We deploy 35 static nodes based on the three different strategies mentioned in Section 7.3, and simulate SADV under different vehicle densities. The results are shown in Fig.7.12. The bottom curve corresponds to the case where each intersection is deployed with a static node. In the figure, strategy 1, strategy 2, and strategy 3 correspond to the three different node deployment strategies respectively: uniform deployment, high-degree preferred, and high-speed preferred strategies. We can observe that given a specific number of static nodes, placing static nodes at high-degree and high-speed intersections have better effect of reducing data delivery delay than uniform deployment of nodes.

7.5 Summary

As the vehicular network is completely composed of mobile nodes, the multi-hop data delivery performance may degrade under median or low vehicle densities where the network is frequently disconnected. In this chapter, we present SADV, which reduces the data delivery delay by three mechanisms. In SNAR, a packet is forwarded to the static node if there are no vehicles available to deliver the packet along the optimal path at the moment, and thus the packet can wait in the static node till the optimal packet delivery path becomes available. In LDU, the adjacent static nodes measure the link delay between each other in real time, so that the routing decision can be made adaptive to the changing vehicle densities. In addition, MPDD can be used to further decrease the packet delivery delay by trying to hit a faster delivery path. Our simulation results have shown that SADV decreases the data delivery delay by around 60 percent under median and low vehicle densities as compared with VADD, where no static nodes are used to assist data routing. Furthermore, our protocol can also adapt to the case of partial deployment of static nodes. We have demonstrated that some intelligent deployment strategies perform better than the random deployment strategy.

CHAPTER 8

Conclusions and Future Work

In this chapter, we conclude the work presented in this dissertation and discuss some future study.

8.1 Conclusions

In the previous discussion, we have demonstrated that there are two challenges for the design of multi-hop wireless networks. The first challenge is the resource allocation. Due to their unique hardware characteristics, different networks have different critical resources, such as the energy in wireless sensor networks and the channel in wireless mesh networks. The second challenge is routing, which is usually coupled with resource allocation. As different networks are intended to support different applications, there are different requirements on the design of routing algorithms, such as increasing network throughput or reducing data delivery delay. In this dissertation, we have discussed several research topics that address these challenges.

The design of energy-efficient algorithms and protocols is critical for wireless sensor networks to prolong the network lifetime. Unlike previous approaches that schedule the sleeping of sensor nodes based on geographic partition, we proposed to partition the nodes based on their measured connectivity. We formulated it as a constrained

optimal graph partition problem, and presented an efficient distributed partitioning algorithm CPA to approximate a good partition. We have demonstrated that CPA outperforms other approaches in two aspects. First, CPA can guarantee K-vertex connectivity of the backbone network under ideal radio propagation models. In addition, CPA ensures K-vertex connectivity of the backbone network with high probability under irregular radio propagation models. Therefore, CPA has better adaptivity in real radio environments.

In multi-channel wireless mesh networks, the appropriate assignment of channels plays an important role in improving the network capacity, and therefore providing users with higher quality of services. We have investigated both static and hybrid channel assignment for wireless mesh networks.

- For static channel allocation, we have demonstrated that the use of partially overlapping channels can help improve the network throughput. We formulated the problem of optimal channel assignment with partially overlapping channels, and presented two channel assignment algorithms, the greedy algorithm and the genetic algorithm. The greedy algorithm is fast but may not get good results, while the genetic algorithm has the potential to get better results with polynomial running time. When the same algorithm is used, the network throughput can be dramatically improved by using all channels compared to using non-overlapping channels only, because the network interference can be better mitigated. When the same set of channels are used, the genetic algorithm excels the previous algorithms, because it is able to solve for better solutions with less interference.

- Considering the inflexibility of static channel assignment and the overhead of dynamic channel switching, we proposed a hybrid wireless mesh network architecture, where each mesh node has both static and dynamic interfaces. We discussed how to coordinate the channel assignment of both static and dynamic

interfaces. To reduce the data delivery delay, we proposed an adaptive dynamic channel allocation algorithm to be used on dynamic interfaces, which enables the negotiation of channels across two hops instead of only one hop in each time interval. In addition, we proposed an interference and congestion aware routing algorithm, which balances the channel usage and therefore improves the network throughput in the network. The hybrid architecture outperforms both pure static and dynamic channel assignment approaches.

As wireless mesh network is expected to be used as infrastructure network to support high performance applications, such video streaming applications, the routing usually requires high-quality paths. We have proposed a multi-path routing algorithm for multi-source VoD applications, which determines both the paths and the video rate on each path. We presented two heuristic path discovery algorithms to find multiple independent and high-quality paths for each VoD request. The algorithms consider both edge-disjointness and interference-disjointness, so they are able to balance the video streaming traffic both spatially and on different channels in the network. Based on the multiple paths discovered, we also proposed a rate allocation algorithm to determine the rate on each path. Our routing algorithm aims at minimizing the network congestion caused by each new video session. Therefore, it not only improves the video streaming quality of each session, but also increases the maximum number VoD users that can be supported in the network.

The design of efficient routing algorithms in vehicular network is a challenging task, because it is highly mobile and frequently disconnected. To improve the data delivery delay of vehicular networks under median or low vehicle densities, we proposed a static node assisted routing protocol SADV. With static nodes deployed at road intersections, a packet is forwarded to the static node when necessary, and the static node chooses the appropriate time to continue to forward the packet to vehicles. The algorithm was further enhanced by real-time link delay update and multi-path

data delivery. We have demonstrated that SADV improves the data delivery delay without causing dramatic increase in protocol overhead, compared to previous work. In cases when we can only have partial deployment of static nodes in the road map, we proposed some efficient intelligent node deployment strategies as well.

8.2 Future Work

In this section, we introduce some potential future work in this area of research.

8.2.1 Spectrum Allocation in Cognitive Radio Networks

Cognitive Radio (CR) or Software Defined Radio (SDR) is envisioned to be the key enabling radio technology of next generation wireless networks [8]. Today's wireless networks operate within fixed spectrum, such as Wi-Fi, Bluetooth, TV and so on. However, a considerable portion of the licensed spectrum is under-utilized with the variation of time and geographic locations. In a typical cognitive radio network, the cognitive radio is able to sense the free channels across both unlicensed and licensed bands over a wide range of spectrum, and switch to any available channel. While cognitive radio is more powerful than the traditional radio, it also has several limitations, which arouses some potential research topics.

1. Spectrum Mobility

Although cognitive radio enjoys more spectrum resource, it must ensure that it will not interfere with any primary users in the licensed bands. If the cognitive radio detects that some primary users begin to use the licensed channels it is currently operating on, it must give up these channels. This overhead, which includes evacuating the channel, sensing new available channels, switching channels, and setting up the link, is non-negligible, and totally takes hundreds of milliseconds [114]. The effect is

called spectrum mobility.

One prospective way of reducing the overhead is to let cognitive radios work on channels with lower probability of being reclaimed by primary users. While we assign channels to cognitive radios to minimize the network interference or satisfy the traffic load, we also want to make sure that the channels used by cognitive radios has minimum probability to be reclaimed by primary users. In this way, we reduce the chances of spectrum mobility, and therefore reduce the overhead.

2. Spectrum Prediction

Another problem with cognitive radio networks is that the sensing overhead cannot be ignored. The current cognitive radio device can only sense a small fraction of spectrum at a time. Thus, to sense a large range, the device has to split the spectrum range into many pieces, and scan one piece at a time. This causes non-negligible delay for the cognitive radio to discover a new spectrum hole when its originally allocated spectrum is reclaimed by some primary users.

One prospective way of solving this problem is to incorporate the prediction technique. If the radio can scan the spectrum piece that has higher probability of being vacant first, then it can find a new available spectrum faster. The problem becomes more complicated when there are multiple cognitive radios trying to find available spectrums; each cognitive radio needs to determine the spectrum to work on while keeping free of interference from the other radios. In this case, we need to determine the scanning order of the spectrum pieces on each radio, with the goal of minimizing the delay of finding interference-free spectrum allocation for all radios.

8

T

d

u

n

r

pe

Tab

fin

Diff

8.2.2 High Performance Applications in Cognitive Mesh Networks

The wireless mesh networks built by traditional radios usually has limited throughput due to the wireless interference and the limited channel resources. Fortunately, the use of cognitive radio has the potential to alleviate this problem. Cognitive radio is more powerful and flexible than the traditional radio used in multi-channel multi-radio wireless mesh networks [47].

- CR can utilize both unlicensed bands and unused licensed bands, and therefore a CR network can enjoy more spectrum resource.

- Traditional radio can only use fixed channels. For example, each channel of 802.11b/g has around $22M$ frequency bandwidth. In contrast, CR has the flexibility to adjust the center frequency as well as the width of frequency band. Moreover, by using Discontiguous Orthogonal Frequency Division Multiplexing ($DOFDM$) [21], a CR is able to work on non-contiguous spectrum fragments, which makes it more flexible in channel allocation.

- Traditional radio has non-negligible channel switching overhead. CR is targeted for switching of channels on packet level, and thus it is suitable for dynamic channel allocation.

As wireless mesh networks become more powerful with cognitive radios, we expect there will be growing demand of high performance applications in the network. Take multi-source VoD streaming as an example. For each VoD session, we need to find appropriate paths that will satisfy its requirement on quality of service (QoS). Different from the traditional wireless mesh networks,

- Cognitive radio can switch channel dynamically with much lower overhead than traditional radio. Therefore, we should not only find multiple paths for routing,

but also consider the channel allocation on each path to satisfy the session's QoS requirement. By considering routing and channel allocation jointly, we achieve better optimization of the application performance.

- The effect of spectrum mobility may affect the video streaming quality. When a primary user reclaims a channel, it may affect one or more streaming paths of the VoD session. Users do not want the session to be discontinued. It would be better if there are still some paths unaffected to continue the video streaming. As a result, we need to minimize the probability that all streaming paths are affected when a primary user reclaims a certain channel.

Therefore, it is a potential research area to study the joint routing and spectrum allocation schemes to optimize the high performance applications in the cognitive wireless mesh networks.

BIBLIOGRAPHY

[1] Dedicated Short Range Communications (DSRC) home.

[2] http://trace.eas.asu.edu/.

[3] http://www.cc.gatech.edu/fac/ellen.zegura/graphs.html.

[4] http://www.census.gov/geo/www/tiger/.

[5] http://www.inf.ethz.ch/~kasten/research/bathtub/energy_consumption.html.

[6] Hyacinth: An ieee 802.11-based multi-channel wireless mesh network.

[7] www.ee.udel.edu/ bohacek/classes/sensornets2007/meshtutorial.ppt.

[8] I. F. Akyildiz, W.-Y. Lee, M. C. Vuran, and S. Mohanty. Next generation / dynamic spectrum access / cognitive radio wireless networks: A survey. In *Comput. Networks*, 2006.

[9] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci. A survey on sensor networks. In *IEEE Communications Magazine*, 2002.

[10] I. F. Akyildiz, X. Wang, and W. Wang. Wireless mesh networks: A survey. In *Computer Networks*, 2004.

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

[12] 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*, 2004.

[13] S. Banerjee and S. Khuller. A clustering scheme for hierarchical control in multi-hop wireless networks. In *INFOCOM*, 2001.

[14] A. Begen, Y. Altunbasak, and O. Ergun. Multi-path selection for multiple description encoded video streaming. In *ICC*, 2003.

[15] M. Bercachi, P. Collard, M. Clergue, and S. Verel. Studying the effects of dual coding on the adaptation of representation for linkage in evolutionary algorithms. In *Linkage in Evolutionary Computation, Volume 157*, 2008.

[16] J. Burgess, B. Gallagher, D. Jensen, and B. N. Levine. Maxprop: Routing for vehicle-based disruption-tolerant networking. In *INFOCOM*, 2006.

[17] B. Burns, O. Brock, and B. Levine. MV Routing and Capacity Building in Disruption Tolerant Networks. In *INFOCOM*, 2005.

[18] J.-H. Chang and L. Tassiulas. Energy conserving routing in wireless ad-hoc networks. In *INFOCOM*, 2000.

[19] I. Chatzigiannakis, S. Nikoletseas, and P. Spirakis. An Efficient Communication Strategy for Ad-Hoc Mobile Networks. In *DISC*, 2001.

[20] B. Chen, J. K., B. H., and M. R. Span: An energy-efficient coordination algorithm for topology maintenance in ad hoc wireless networks. In *MobiCom*, 2001.

[21] D. Chen, Q. Zhang, and W. Jia. Aggregation aware spectrum assignment in cognitive ad-hoc networks. In *CrownCom*, 2008.

[22] Z. Chen, H. Kung, and D. Vlah. Ad Hoc Relay Wireless Networks over Moving Vehicles on Highways. In *MobiHoc*, 2001.

[23] P. A. Chou and Z. Miao. Rate-distortion optimized streaming of packetized media. In *IEEE Transactions on Multimedia*, 2006.

[24] D. Culler, D. Estrin, and M. Srivastava. Overview of sensor networks. In *Sensor Networks*, 2004.

[25] B. Das and V. Bharghavan. Routing in ad-hoc networks using minimum connected dominating sets. In *ICC*, 1997.

[26] J. A. Davis, A. H. Fagg, and B. N. Levine. Wearable computers as packet transport mechanisms in highly-partitioned ad-hoc networks. In *Proceedings of the 5th IEEE International Symposium on Wearable Computers*, 2001.

[27] B. Deb and B. Nath. On the node-scheduling approach to topology control in ad hoc networks. In *MobiHoc*, 2005.

[28] G. V. der Auwera, P. T. David, and M. Reisslein. Traffic and quality characterization of single-layer video streams encoded with h.264/mpeg-4 advanced video coding standard and scalable video coding extension. In *IEEE Transactions on Broadcasting, Volume 54, Issue 3*, 2008.

[29] G. V. der Auwera and M. Reisslein. Implications of smoothing on statistical multiplexing of h.264/avc and svc video streams. In *IEEE Transactions on Broadcasting, Volume 55, Issue 3*, 2009.

[30] A. Dhananjay, H. Zhang, J. Li, and L. Subramanian. Practical, distributed channel assignment and routing in dual-radio mesh networks. In *SIGCOMM*, 2009.

[31] Y. Ding, Y. Huang, G. Zeng, and L. Xiao. Channel assignment with partially overlapping channels in wireless mesh networks. In *WICON*, 2008.

[32] Y. Ding, K. Pongaliur, and L. Xiao. Hybrid multi-channel multi-radio wireless mesh networks. In *IWQoS*, 2009.

[33] Y. Ding, C. Wang, and L. Xiao. A connectivity based partition approach for node scheduling in sensor networks. In *DCOSS*, 2007.

[34] Y. Ding, C. Wang, and L. Xiao. A static-node assisted adaptive routing protocol in vehicular networks. In *VANET*, 2007.

[35] Y. Ding, C. Wang, and L. Xiao. An adaptive partitioning scheme for sleep scheduling and topology control in wireless sensor networks. In *TPDS, Volume 20, Issue 9*, 2009.

[36] O. Dousse, P. Mannersalo, and P. Thiran. Latency of wireless sensor networks with uncoordinated power saving mechanisms. In *MobiHoc*, 2004.

[37] R. Draves, J. Padhye, and B. Zill. Comparison of routing metrics for static multi-hop wireless networks. In *SIGCOMM*, 2004.

[38] P. Dutta, S. Jaiswal, and R. Rastogi. Routing and channel asslocation in rural wireless mesh networks. In *INFOCOM*, 2007.

[39] E. Falkenauer. The worth of uniform crossover. In *Proceedings of the Congress on Evolutionary Computation*, 1999.

[40] T. Feder, P. Hell, S. Klein, and R. Motwani. Complexity of graph partition problems. In *ACM STOC*, 1999.

[41] R. Friedman and G. Korland. Timed grid routing (tigr) bites off energy. In *MobiHoc*, 2005.

[42] B. K. A. G. Gang Lu, Narayanan Sadagopan. Delay efficient sleep scheduling in wireless sensor networks. In *INFOCOM*, 2005.

[43] A. Graf. Distance graphs and the t-coloring problem. In *Discrete Math*, 1999.

[44] J. Grefenstette. Optimization of control parameters for genetic algorithms. In *IEEE Trans. Syst. Man Cybern., Vo.16, No.1*, 1986.

[45] S. Guha and S. Khuller. Approximation algorithms for connected dominating sets. In *ESA*, 1996.

[46] T. He, C. Huang, B. M. Blum, J. A. Stankovic, and T. F. Abdelzaher. Range-free localization schemes in large scale sensor networks. In *MobiCom*, 2003.

[47] Y. T. Hou, Y. Shi, and H. D. Sherali. Optimal spectrum sharing for multi-hop software defined radio networks. In *INFOCOM*, 2007.

[48] X. Hu and E. D. Paolo. An efficient genetic algorithm with uniform crossover for air traffic control. In *Journal of Computers and Operations Research, Volume 36, Issue 1*, 2009.

[49] S. Irani. Coloring inductive graphs on-line. In *Journal of Algorithmica*, 1994.

[50] J. Tang, G. Xue, and W. Zhang. Interference-aware topology control and qos routing in multi-channel wireless mesh networks. In *MobiHoc*, 2005.

[51] K. Jain, J. Padhye, V. Padmanabhan, and L. Qiu. Impact of interference on multi-hop wireless network performance. In *MobiCom*, 2003.

[52] M. Jerbi, R. Meraihi, S. Senouci, and Y. Chamri-Doudane. Gytar: improved greedy traffic aware routing protocol for vehicular ad hoc networks in city environments. In *VANET*, 2006.

[53] D. B. Johnson and D. A. Maltz. Dynamic source routing in ad hoc wireless networks. In *Mobile Computing*, 1996.

[54] P. Juang, H. Oki, Y. Wang, M. Martonosi, L. Peh, and D. Rubenstein. Energy-efficient computing for wildlife tracking: Design tradeoffs and early experiences with zebranet. In *ASPLOS*, 2002.

[55] B. Karp and H. Kung. GPSR: Greedy Perimeter Stateless Routing for Wireless Networks. In *MobiCom*, 2000.

[56] M. Kodialam and T. Nandagopal. Characterizing the capacity region in multi-radio multi-channel wireless mesh networks. In *MobiCom*, 2005.

[57] S. Kompella, S. Mao, Y. T. Hou., and H. D. Sherali. Cross-layer optimized multipath routing for video communications in wireless networks. In *Journal on Selected Areas in Communications, Volume 25*, 2007.

[58] J. F. Kurose and K. W. Ross. *Computer Networking: A Top-Down Approach Featuring the Internet.* Addison Wesley, 2004.

[59] P. Kyasanur and N. Vaidya. Routing and link-layer protocols for multichannel multi-interface ad hoc wireless networks. In *ACM MC2R*, 2006.

[60] K. Lee, M. Le, J. Haerri, and M. Gerla. Louvre: Landmark overlays for urban vehicular routing environments. In *Proceedings of IEEE WiVeC*, 2008.

[61] S. J. Lee and M. Gerla. Split multipath routing with maximally disjoint paths in ad hoc networks. In *ICC*, 2001.

[62] D. Li, Q. Zhang, C.-N. Chuah, and S. J. B. Yoo. Multi-source multi-path video streaming over wireless mesh networks. In *Proc. ISCAS*, 2006.

[63] N. Li and J. C. Hou. Flss: A fault-tolerant topology control algorithm for wireless networks. In *MobiCom*, 2004.

[64] Q. Li and D. Rus. Sending messages to mobile users in disconnected ad-hoc wireless networks. In *Mobile Computing and Networking*, 2000.

[65] X.-Y. Li, W.-Z. Song, and W. Wang. A unified energy-efficient topology for unicast and broadcast. In *MobiCom*, 2005.

[66] C. Lochert, M. Mauve, H. Fler, and H. Hartenstein. Geographic routing in city scenarios. In *ACM SIGMOBILE Mobile Computing and Communications Review, Vol.9, No.1*, 2005.

[67] J. Luo and J.-P. Hubaux. Joint mobility and routing for lifetime elongation in wireless sensor networks. In *INFOCOM*, 2005.

[68] S. Mao, Y. T. Hou, X. Cheng, H. D. Sherali, and S. F. Midkiff. Multipath routing for multiple description video in wireless ad hoc networks. In *INFOCOM*, 2005.

[69] S. Mao, S. Kompella, Y. Hou, H. Sherali, and S. Midkiff. Routing for multiple concurrent video sessions in wireless ad hoc networks. In *ICC*, 2005.

[70] M. Marina and S. Das. On-demand multipath distance vector routing in ad hoc networks. In *ICNP*, 2001.

[71] T. M. Michell. *Machine Learning.* 1997.

[72] A. Mishra, S. Banerjee, and W. Arbaugh. Weighted coloring based channel assignment for wlans. In *MC2R*, 2005.

[73] A. Mishra, E. Rozner, S. Banerjee, and W. Arbaugh. Exploiting partially overlapping channels in wireless networks: Turning a peril into an advantage. In *Internet Measurement Conference*, 2005.

[74] A. Mishra, V. Shrivastava, S. Banerjee, and W. A. Arbaugh. Partially overlapped channels not considered harmful. In *SIGMETRICS*, 2006.

[75] V. Navda, A. Kashyap, S. Ganguly, and R. Izmailov. Real-time video stream aggregation in wireless mesh network. In *Proc. PIMRC*, 2006.

[76] T. Nguyen and A. Zakhor. Distributed video streaming with forward error correction. In *Packet Video Workshop*, 2002.

[77] T. Nguyen and A. Zakhor. Multiple sender distributed video streaming. In *IEEE Transactions on Multimedia, Volume 6, Issue 2*, 2004.

[78] T. P. Nguyen and A. Zakhor. Distributed video streaming over internet. In *Proc. SPIE, Multimedia Computing and Networking*, 2002.

[79] D. Niculescu and B. Nath. Trajectory Based Forwarding and Its Applications. In *MobiCom*, 2003.

[80] M. R. Pearlman, Z. J. Haas, P.Sholander, and S. S. Tabrizi. On the impact of alternate path routing for load balancing in mobile ad hoc networks. In *MobiHoc*, 2000.

[81] S. Pediaditaki, P. Arrieta, and M. K. Marina. A learning-based approach for distributed multi-radio channel allocation in wireless mesh networks. In *ICNP*, 2009.

[82] M. D. Penrose. On k-connectivity for a geometric random graph. In *Wiley Random Structures and Algorithms*, 1999.

[83] C. Perkins, E. Royer, and S. Das. Ad hoc on demand distance vector (AODV) routing. draft-ietf-manet-aodv-10.txt. Internet Draft, IETF, 2002.

[84] K. F. Pl. Genetic algorithm with local optimization. In *Journal of Biological Cybernetics, Volume 73, Issue 4*, 1995.

[85] R. Draves, J. Padhye and B. Zill. Routing in multi-radio multi-hop wirelss mesh networks. In *MobiCom*, 2004.

[86] 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.

[87] B. Raman. Channel allocation in 802.11-based mesh networks. In *INFOCOM*, 2006.

[88] B. Raman and K. Chebrolu. Design and evaluation of a new mac protocol for long-distance 802.11 mesh networks. In *MobiCom*, 2005.

[89] R. Ramanathan and R. Hain. Topology control of multihop wireless networks using transmit power adjustment. In *INFOCOM*, 2000.

[90] A. Raniwala and T. cker Chiueh. Architecture and algorithms for an ieee 802.11-based multi-channel wireless mesh network. In *INFOCOM*, 2005.

[91] A. Raniwala, K. Gopalan, and T. cker Chiueh. Centralized channel assignment and routing algorithms for multi-channel wireless mesh networks. In *MC2R*, 2004.

[92] A. K. Saha and D. B. Johnson. Modeling Mobility for Vehicular Ad Hoc Networks. In *VANET*, 2004.

[93] A. Sankar and Z. Liu. Maximum lifetime routing in wireless ad-hoc networks. In *INFOCOM*, 2004.

[94] J. D. Schaffer, R. Caruana, L. J. Eshelman, and R. Das. A study of control parameters affecting online performance of genetic algorithms for function optimization. In *Proceedings of the 3rd International Conference on Genetic Algorithms*, 1989.

[95] P. Seeling, M. Reisslein, and B. Kulapala. Network performance evaluation with frame size and quality traces of single-layer and two-layer video: A tutorial. In *IEEE Communications Surveys and Tutorials, Volume 6, No.3*, 2004.

[96] J. So and N. Vaidya. Multi-channel mac for ad hoc networks: Handling multi-channel hidden terminals using a single transceiver. In *MobiHoc*, 2004.

[97] M. Stemm and R. H. Katz. Measuring and reducing energy consumption of network interfaces in hand-held devices. In *IEICE TC*, 1997.

[98] A. P. Subramaniam, H. Gupta, and S. R. Das. Minimum-interference channel assignment in multi-radio wireless mesh networks. In *SECON*, 2007.

[99] A. P. Subramaniam, H. Gupta, and S. R. Das. Minimum-interference channel assignment in multi-radio wireless mesh networks. In *SECON*, 2007.

[100] R. Subramanian and F. Fekri. Sleeping scheduling and lifetime maximization in sensor networks: Fundamental limits and optimal solutions. In *IPSN*, 2006.

[101] W.-H. Tam and Y.-C. Tseng. Joint multi-channel link layer and multi-path routing design for wireless mesh networks. In *INFOCOM*, 2007.

[102] D. Tian and N. Georganas. A coverage-preserving node scheduling scheme for large wireless sensor networks. In *ACM WSNA-MOBICOM*, 2003.

[103] J. Tian, L. Han, K. Rothermel, and C. Cseh. Spatially Aware Packet Routing for Mobile Ad Hoc Inter-vehicle Radio Networks. In *Proc. of the IEEE 6th Intl. Conf. on Intelligent Transportation Systems (ITSC)*, 2003.

[104] D. A. Tran, K. A. Hua, and T. T. Do. A peer-to-peer architecture for media streaming. In *Journal on Selected Areas in Communications, Volume 22*, 2003.

[105] A. Vahdat and D. Becker. Epidemic routing for partially connected ad hoc networks. Technical report, 2000.

[106] T. van Dam and K. Langendoen. An adaptive energy-efficient mac protocol for wireless sensor networks. In *SenSys*, 2003.

[107] W. Wang, V. Srinivasan, and K.-C. Chua. Using mobile relays to prolong the lifetime of wireless sensor networks. In *MobiCom*, 2005.

[108] W. Wang, F. Xie, and M. Chatterjee. Small-scale and large-scale routing in vehicular ad hoc networks. In *IEEE Transactions on Vehicular Technology, Vol.58, No.9*, 2009.

[109] R. Wattenhofer, L. Li, P. Bahl, and Y.-M. Wang. Distributed topology control for wireless multihop ad-hoc networks. In *INFOCOM*, 2001.

[110] W. Wei and A. Zakhor. Path selection for multi-path streaming in wireless ad hoc networks. In *Proc. ICIP*, 2006.

[111] H. Wu, R. Fujimoto, R. Guensler, and M. Hunter. MDDV: Mobility-Centric Data Dissemination Algorithm for Vehicular Networks. In *Proceedings of VANET*, 2004.

[112] J. Wu, F. Dai, M. Gao, and I. Stojmenovic. On calculating power-aware connected dominating sets for efficient routing in ad hoc wireless networks. In *JCN*, 2002.

[113] S.-L. Wu, C.-Y. Lin, Y.-C. Tseng, and J.-P. Sheu. A new multi-channel mac protocol with on-demand channel assignment for multi-hop mobile ad hoc networks. In *ISPAN*, 2000.

[114] D. Xu, E. Jung, and X. Liu. Optimal bandwidth selection in multi-channel cognitive radio networks: How much is too much? In *DySPAN*, 2008.

[115] Q. Xu, T. Mark, J. Ko, and R. Sengupta. Vehicle-to-vehicle Safety Messaging in DSRC. In *VANET*, 2004.

[116] Y. Xu and et al. Topology control protocols to conserve energy in wireless ad hoc networks. Technical report, 2003.

[117] Y. Xu, J. Heidemann, and D. Estrin. Geography-informed energy conservation for ad hoc routing. In *MobiCom*, 2001.

[118] W. Ye, J. Heidemann, and D. Estrin. An energy-efficient mac protocol for wireless sensor networks. In *INFOCOM*, 2002.

[119] J. Yin, T. ElBatt, G. Yeung, B. Ryu, S. Habermas, H. Krishnan, and T. Talty. Performance Evaluation of Safety Applications over DSRC Vehicular Ad Hoc Networks. In *VANET*, 2004.

[120] J. Yoshino and I. Ohtomo. Study on efficient channel assignment method using the genetic algorithm for mobile communication systems. In *Journal of Soft Computing, Volume 9, Issue 2*, 2005.

[121] H. Yu, D. Zheng, B. Y. Zhao, and W. Zheng. Understanding user behavior in large scale video-on-demand systems. In *Proc. of ACM EuroSys*, 2006.

[122] W. Yue, K. Miyazaki, and X. Deng. Optimal channel assignment in wireless communication networks with distance and frequency interferences. In *Computer Communications*, 2004.

[123] J. Zhao and G. Cao. VADD: Vehicle-Assisted Data Delivery in Vehicular Ad Hoc Networks. In *IEEE Transactions on Vehicular Technology, Vol.57, No.3*, 2008.

[124] W. Zhao, M. Ammar, and E. Zegura. A message ferrying approach for data delivery in sparse mobile ad hoc networks. In *MobiHoc*, 2004.

[125] W. Zhao, Y. Chen, M. Ammar, M. Corner, B. Levine, and E. Zegura. Capacity enhancement using throwboxes in dtns. In *IEEE MASS*, 2006.

[126] X. Zhu, S. Han, and B. Girod. Congestion-aware rate allocation for multipath video streaming over ad hoc wireless networks. In *Proc. ICIP*, 2004.