國立交通大學

資訊科學與工程研究所

博士論文

IEEE 802.16 網路之無線資源管理

Wireless Resource Management in IEEE 802.16 Networks

研究生：陳烈武

指導教授：曾煜棋 教授

中華民國九十七年十二月

IEEE 802.16 網路之無線資源管理
Wireless Resource Management in IEEE 802.16 Networks

研究生：陳烈武　　Student：Lien-Wu Chen

指導教授：曾煜棋　　Advisor：Yu-Chee Tseng

國立交通大學

資訊科學與工程研究所

博士論文

A Dissertation

Submitted to Department of Computer Science

College of Computer Science

National Chiao Tung University

in partial Fulfillment of the Requirements

for the Degree of

Doctor of Philosophy

in

Computer Science

December 2008

Hsinchu, Taiwan, Republic of China

中華民國九十七年十二月

IEEE 802.16 網路之無線資源管理

學生：陳烈武　　　　　　　　　指導教授：曾煜棋教授

國立交通大學資訊工程學系（研究所）博士班

摘　　要

無線城域網路之國際標準 IEEE 802.16 已被定義出來滿足低成本的大範圍寬頻無線存取(broadband wireless access)，在本篇論文中，我們將充分開發頻譜再利用(spectral reuse)和競爭碰撞解決(contention resolution)之可能性來進一步提昇無線頻寬的使用效率。本篇論文內容分為傳輸排程(scheduling)、封包繞徑(routing)、以及頻寬要求(requesting)三大議題

在第一個研究主題中，我們深入地研究如何在 IEEE 802.16 網狀網路(mesh network)的資源配置(resource allocation)中充分開發出頻譜再利用，其中包括繞徑樹建構(routing tree construction)、頻寬配置(bandwidth allocation)、時槽分派(time-slot assignment)、以及即時資料流(real-time flow)的頻寬保證(bandwidth guarantee)。我們提出的頻譜再利用 framework 涵蓋了應用層(application layer)的頻寬配置、媒體存取層(MAC layer)的繞徑樹建構與資源分享、以及實體層(physical layer)的頻道再利用(channel reuse)。就目前所知，這是第一個研究成果以數學方式分析出 IEEE 802.16 網狀網路的頻譜再利用程度，並且設計出完整的 framework 來提昇頻譜再利用的效率。模擬結果顯示我們所提出之 framework 大幅度地增加了 IEEE 802.16 網狀網路的網路產出量(network throughput)。

在第二個研究主題中，當 IEEE 802.16 網狀網路的網狀傳輸站(mesh station)具備移動能力之後，便形成一個由行動中繼傳輸站(mobile relay station)

所構成的行動隨意網路(mobile ad hoc networks)，行動隨意網路由於其極具彈性的網路架構，已經廣受各方的注目，雖然有許多根據不同準則而為行動隨意網路所設計的繞徑協定(routing protocol)，但是其中只有極少數考量到已被許多無線網路設備所支援的多重速率傳輸(multi-rate)之影響。在給定一條傳輸路徑(routing path)的情況下，我們提供了一套數學分析的工具，假設行動中繼傳輸站以離散時間隨機方式(discrete-time, random-walk)來移動，進而計算出此傳輸路徑的期望產出量(expected throughput)，並且將頻譜再利用的因素一併考量進來。模擬結果顯示我們所提出之數學分析方法可準確地計算出傳輸路徑的期望產出量，其推導結果可以作為更佳的傳輸路徑選擇(route selection)準則。

在第三個研究主題中，為了更有效率地使用無線資源，我們進一步地研究了兩個在 IEEE 802.16 網路中針對 best-effort traffics 解決頻寬要求(bandwidth request)碰撞的機制。其中一個是定義在標準中的 exponential backoff 機制，另外一個是我們所提出 single-frame backoff 的 piggyback 機制。我們分析並比較了這兩個機制在 Poisson traffic 下頻寬要求成功機率(request success probability)和封包傳輸延遲(packet delivery delay)方面的效能。分析和模擬結果顯示 piggyback 機制的效能表現比 exponential backoff 要來得出色許多，可以大幅度地減低頻寬要求的碰撞。

關鍵字：競爭碰撞解決、IEEE 802.16、媒體存取控制、行動計算、資源配置、網狀網路、封包繞徑、頻譜再利用、WiMAX、無線通訊

Wireless Resource Management in IEEE 802.16 Networks

Student: Lien-Wu Chen Advisor: Prof. Yu-Chee Tseng

Department of Computer Science

National Chiao Tung University

ABSTRACT

The IEEE 802.16 standard for wireless metropolitan area networks (WMAN) is defined to meet the need of wide-range broadband wireless access at low cost. In this dissertation, we exploit spectral reuse and contention resolution of IEEE 802.16 networks. This dissertation is composed of three works. In the first work, we exploits spectral reuse in an IEEE 802.16 mesh network through bandwidth allocation, time-slot assignment, and routing tree construction. In the second work, we provides an analytic tool to evaluate the expected throughput of the route with spectral reuse in an IEEE 802.16 relay network. To further improve wireless resource utilization, the last work analyzes and compares two collision-resolution requesting schemes for best-effort (BE) traffics in IEEE 802.16 networks.

In this dissertation, we first study how to exploit spectral reuse in resource allocation in an IEEE 802.16 mesh network, which includes routing tree construction, bandwidth allocation, time-slot assignment, and bandwidth guarantee of real-time flows. The proposed spectral reuse framework covers bandwidth allocation at the application layer, routing tree construction and resource sharing at the MAC layer, and channel reuse at the physical layer. To the best of our knowledge, this is the first effort which formally quantifies spectral reuse in IEEE 802.16 mesh networks and which exploits spectral efficiency under an integrated framework. Simulation results show that the proposed schemes significantly improve the throughput of IEEE 802.16 mesh networks.

On the other hand, when mesh stations have mobility, they form a mobile ad hoc network (MANET) consisted of mobile relay stations. While many routing protocols have been proposed for MANETs based on different criteria, few have considered the impact of multi-rate communication capability that is supported by many current wireless products. Given a routing path, the second work provides an analytic tool to evaluate the expected throughput of the route with spectral reuse in a mobile relay network, assuming that hosts move following the discrete-time, random-walk model. The derived result can be added as another metric for route selection. Simulation results show that the proposed formulation can be used to evaluate path throughput accurately.

To utilize the channel bandwidth more efficiently, the third work studies two collision-resolution requesting schemes for best-effort (BE) traffics in IEEE 802.16 networks. One is the exponential backoff scheme defined in the standard and the other is a piggyback mechanism enhanced by single-frame backoff, called the *Request Piggyback* (RPB) scheme. We analyze and compare their performance in terms of the request success probability and the packet delivery delay under Poisson traffic. The results show that the RPB scheme outperforms the exponential backoff scheme and can reduce request collision. Based on the designed scheduling, routing, and requesting schemes, we can further improve the efficiency of wireless resource management in IEEE 802.16 Networks.

Keywords: contention resolution, IEEE 802.16, medium access control, mobile computing, resource allocation, mesh network, routing, spectral reuse, WiMAX, wireless communication.

Acknowledgement

Special thanks go to my advisor Prof. Yu-Chee Tseng for his guidance in my dissertation work. I would also like to thank my dissertation committee members: Prof. Chien-Chao Tseng, Prof. Shiao-Li Tsao, Prof. Jan-Jan Wu, Prof. Da-Wei Wang, Prof. Kun-Ming Yu, and Prof. Tong-Ying Juang. They asked me some good questions and gave me useful comments so that I can improve my work in the future.

Let me also say thank to those HSCC members who co-work with me and all guys I meet in NCTU. Because of you, I can have a great time during these years. Finally, I will dedicate this dissertation to my families and Ms. Chen who is my girl friend in the past, my wife in the present, and my children's mother in the future.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

To achieve the requirement of wide-range wireless broadband access at a low cost, the IEEE 802.16 standard [4] has been proposed recently. The goal of this standard is to solve the last-mile problem in a metropolitan area network in a more flexible and economical way as opposed to traditional cabled access networks, such as fiber optics, DSL (digital subscriber line), or T1 links [5, 6]. The IEEE 802.16 standard is based on a common MAC (medium access control) protocol compliant with different physical layer specifications. The physical layer can employ the OFDM (orthogonal frequency division multiplexing) scheme below 11 GHz or the single carrier scheme between 10 GHz and 66 GHz.

The IEEE 802.16 MAC protocol supports the *point-to-multipoint (PMP)* mode and the *mesh* mode. In the PMP mode, stations are organized as a cellular network, where subscriber stations (SSs) are directly connected to base stations (BSs). Such networks require each SS to be within the communication range of its associated BS, thus greatly limiting the coverage range of the network. On the other hand, in the mesh mode, stations are organized in an ad-hoc fashion. Each SS can either act as an end point or a router to relay traffics for its neighbors. Thus, there is no need to have a direct link from each SS to its associated BS. This leads to two advantages: SSs may transmit at higher rates to their parent SSs or BS, and a BS can serve wider coverage at a lower deployment cost [7].

In this dissertation, we first study the spectral reuse issue in an IEEE 802.16 mesh network through multi-hop routing and scheduling, while there is no spectral reuse considered in the IEEE 802.16 standard. The proposed framework includes a load-aware routing algorithm and a centralized scheduling scheme, which consider both bandwidth demands and interference among SSs. Given traffic patterns of SSs, we show how to achieve better spatial reuse and thus higher spectral efficiency.

On the other hand, when mesh stations have mobility, they form a mobile ad hoc network (MANET) consisted of mobile relay stations. The MANET is a flexible and dynamic architecture that is attractive due to its ease in network deployment. Routing is perhaps one of the most intensively addressed issues in MANET. Many different criteria have been used in route selection, including hop count [8], signal strength [9], route lifetime [10], and energy constraint [11]. Among these metrics, hop count may be the most widely used metric in choosing routes. When a hop-count based routing protocol is given multiple paths, the shortest path is normally selected and a random path is selected when there is a tie. This metric has the advantage of simplicity, requiring no additional measurements and incurring the least number of transmissions. The primary disadvantage of this metric is that it does not take packet loss or available bandwidth into account, especially when network interfaces can transmit at multiple rates [12]. It has been shown in [13] that a route which minimizes the hop count does not necessarily maximize the throughput of a flow.

While it is true that there is no single route selection metric that is able to best fit all possible routing scenarios in MANET, few works have considered the impact of multi-rate communication capability that is widely supported by many current wireless LAN products. For example, IEEE 802.11b supports rates of 11, 5.5, 2, and 1 Mbps, while IEEE 802.11a supports rates of 6, 9, 12, 18, $\cdots$, and 54 Mbps. Route selection is more complicated in a multi-rate MANET than in a single-rate environment. Also, there exists an inherent tradeoff between transmission rates and their effective transmission ranges [14]. To support reliable data transmissions, longer-range communications must use lower rates, and vice versa. Auto-rate selection protocols [15, 16] do exist at the link level. Reference [17] proposes a multi-rate-aware topology control algorithm to enhance the network throughput in multi-hop ad hoc networks, and [18] uses fast links (with high nominal bit rate) to improve the system throughput in wireless mesh networks. However, they only focus on static network environment without taking mobility into account. Reference [19] proposes a multi-rate-aware sub-layer between the MAC and the network layers to improve resource utilization and to minimize power consumption, but the effect of multi-rate communications at the routing level is not yet fully addressed.

In the second work, we consider a MANET consisted of mobile relay stations where each wireless link can support multiple rates and has the auto-rate selection capability. Given a routing path, this work provides an analytic tool to evaluate the expected throughput of the route with spectral reuse, assuming that hosts move following the discrete-time, random-walk model.

To utilize the channel bandwidth more efficiently, we study the centralized, reservation-based bandwidth allocation mechanism defined in IEEE 802.16 for best-effort (BE) traffics. It adopts Time Division Multiplexing (TDM) for the downlink channel and Time Division Multiple Access (TDMA) for the uplink channel via a request/grant mechanism controlled by the BS. The uplink channel is modelled as a stream of time slots. SSs must send request messages to the BS to reserve uplink bandwidth. There are three factors that may affect the performance of the uplink channel: (i) the portion of request slots per frame, (ii) the collision-resolving procedure, and (iii) the allocation of slots to SSs' requests. In the last work, we studies the collision-resolution mechanisms for transmitting uplink BE requests to the BS. The request scheme defined in the standard is compared against the proposed *Request Piggyback* (RPB) scheme.

Chapter 2

Exploiting Spectral Reuse in Routing, Resource allocation, and Scheduling for IEEE 802.16 Mesh Networks

2.1 Observations and Motivations

In this work, we study the spectral reuse issue in an IEEE 802.16 mesh network through multi-hop routing and scheduling, while there is no spectral reuse considered in the IEEE 802.16 standard. The proposed framework includes a load-aware routing algorithm and a centralized scheduling scheme, which consider both bandwidth demands and interference among SSs. Given traffic patterns of SSs, we show how to achieve better spatial reuse and thus higher spectral efficiency.

In an IEEE 802.16 mesh network, transmissions can undergo a multi-hop manner. The standard specifies a centralized scheduling mechanism for the BS to manage the network. Stations will form a *routing tree* rooted at the BS for the communication purpose. SSs in the network will send request messages containing their traffic demands and link qualities to the BS to ask for resources. The BS then uses the topology information along with SSs' requests to determine the routing tree and to allocate resources. Resources in an IEEE 802.16 network are usually represented by time slots within a frame. Our goal is to solve the *resource allocation problem*, given the uplink/downlink bandwidth demands of each SS and their link qualities. There are four issues to be considered:

- Tree reconstruction: How to determine the routing tree based on SSs' current bandwidth demands and link qualities?
- Bandwidth allocation: How to determine the number of time slots of each SS according to its uplink and downlink bandwidth demands?
- Time-slot assignment: How to assign time slots to each SS in a frame?
- Bandwidth guarantee: How to schedule transmission on time slots for each SS, so that a fixed amount of bandwidth is guaranteed for each real-time flow?

In this work, we investigate the resource allocation problem by exploring the concept of *spectral reuse*. Although it is well-known that a time slot used by a station can be "reused" by another station if the latter is sufficiently separated from the former, the IEEE 802.16 standard does not explore in this direction. We propose a spectral reuse framework to efficiently allocate resources in an IEEE 802.16 mesh network with global fairness in mind, that is, the bandwidths allocated to SSs will be proportionate to their requests, in an end-to-end (SS-to-BS) sense. Our framework includes a routing tree construction and a centralized scheduling algorithm. The former allows a BS to form an efficient routing tree according to SSs' bandwidth demands and interferences. The latter helps a BS to determine bandwidth allocation and time-slot assignment. In particular, when time slots are tight, we show how to adjust scheduling to prioritize real-time from non-real-time traffics so as to guarantee some bandwidths for real-time traffics. Note that the tree topology is consistent with the current IEEE 802.16 standard. Also, our framework does not require any change to the message structures and the signaling mechanism defined in the standard.

In the literature, early works on the IEEE 802.16 standard have primarily focused on the PMP mode [20–22]. For the mesh mode, former efforts have devoted to topology design [23],

Table 2.1: Comparison of prior works [1–3] and our spectral reuse framework.

	reuse	load	tree	time-slot	bandwidth
features	modeling[1]	awareness	reconstruction	allocation	guarantee[3]
reference [1]			partial[2]	√	
reference [2]			partial[2]		
reference [3]			√	√	
our framework	√	√	√	√	√

[1] Mathematical modeling is provided to evaluate the degree of spectral reuse.
[2] Initial tree construction is provided, but without tree reconstruction.
[3] The guarantee is for real-time flows.

packet scheduling [24, 25], and QoS support [26, 27]. Reference [28] shows how to manage radio resources in a WiMAX single-carrier network in a distributed manner. Reference [29] discusses how to improve channel efficiency and provide fair access to SSs. The BS allocates time slots to SSs in a per-hop basis in such a way that one-hop nodes will have precedence over two-hop nodes ("hop" in the sense of nodes' distances to the BS). Similarly, i-hop nodes will have precedence over (i+1)-hop nodes. However, this may lead to starvation of farther-away SSs as the network becomes congested, especially when SSs with smaller hop counts request larger bandwidths. On the contrary, our scheduling algorithm allocates time slots to SSs proportionate to their requests and thus avoids such starvation.

Several studies [1–3] have addressed the issue of spectral reuse to solve the resource allocation problem. Reference [1] proposes a routing tree construction and a scheduling algorithm by considering the interference among neighboring SSs. It attempts to find a route to reduce the interference among SSs, and then to maximize the number of concurrent transmissions. How to attach a new SS to a routing tree incurring the least interference is discussed in [2]. In [3], the authors indicate that the network performance highly depends on the order that SSs join the routing tree, and then propose a routing tree reconstruction and a concurrent transmission scheme to achieve spectral reuse. As can be seen, the prior works only discuss partial aspects of the resource allocation problem.

Table 2.1 compares the functions provided by other schemes and ours. Our framework offers the most complete solution to the resource allocation problem. The contributions of our framework are four-fold. First, it formally quantifies the spectral reuse in a mesh network, thus capable of achieving higher spectral efficiency. Second, it takes dynamic traffic demands of SSs into account and includes not only a tree optimization algorithm, but also a bandwidth allocation and a time-slot assignment. Third, we propose a way to prioritize real-time from non-real-time traffics, so that a fixed amount of bandwidth is maintained for each real-time flow when resources are stringent. Finally, the proposed framework covers bandwidth allocation at the application layer, routing tree construction and resource sharing at the MAC layer, and channel reuse at the physical layer. Extensive performance studies are conducted and the simulation results show that our framework can achieve better spectral reuse and higher network throughput compared with existing results.

2.2 Background and Problem Definition

2.2.1 Resource Allocation in an IEEE 802.16 mesh network

An IEEE 802.16 mesh network is composed of a BS and several SSs. These stations form a routing tree rooted at the BS and transmissions between stations may undergo a multi-hop manner. The IEEE 802.16 MAC protocol supports both centralized and distributed scheduling methods. In this work, we focus on the centralized scheduling to fully exploit spectral reuse.

In the centralized scheduling, the standard supports two control messages, *MSH-CSCF (Mesh Centralized Scheduling Configuration)* and *MSH-CSCH (Mesh Centralized Scheduling)*, to help the BS establish its routing tree and specify transmission schedules of SSs in the network. To achieve this, the BS first broadcasts an MSH-CSCF message containing the routing tree information to the network. An SS receiving such a message can know its parent and children in the tree and then rebroadcasts the MSH-CSCF message according to its index specified in the message. This procedure is repeated until all SSs have received the MSH-CSCF message.

After constructing the routing tree by the MSH-CSCF message, SSs can transmit MSH-CSCH:Request messages to request time slots. The transmission order is from leaves to the root. An SS will combine the requests from its children into its own MSH-CSCH:Request message, and then transmits the message to its parent. In this way, the BS can gather bandwidth requests from all SSs and then broadcasts an MSH-CSCH:Grant message containing the slot allocations to all SSs. Note that the BS can also update the routing tree by containing tree update information in the MSH-CSCH:Grant message. In this case, SSs have to update their positions in the new tree according to the message. Otherwise, the routing tree remains the same as specified in the previous MSH-CSCF message. Note that according to the 802.16 standard, the period during which the MSH-CSCH schedule is valid is limited by the time that the BS takes to aggregate traffic requirements and distribute the next schedule. So the scheduling interval is about several frames depending on the size of the mesh network. Therefore, it is reasonable to assume that link data rates and bandwidth demands of SSs are constants during a short period of time.

To allocate bandwidths for SSs, the IEEE 802.16 standard gives an example, as illustrated in Fig. 2.1. Each SS i first sends its uplink bandwidth demand b_i^{UL} and downlink bandwidth demand b_i^{DL} to the BS. Let the uplink and downlink data rates of SS i be r_i^{UL} and r_i^{DL}, respectively. The ratios of uplink slots allocated to SS 1, SS 2, SS 3, and SS 4 will be $\frac{b_1^{\text{UL}}+b_3^{\text{UL}}+b_4^{\text{UL}}}{r_1^{\text{UL}}} : \frac{b_2^{\text{UL}}}{r_2^{\text{UL}}} : \frac{b_3^{\text{UL}}}{r_3^{\text{UL}}} : \frac{b_4^{\text{UL}}}{r_4^{\text{UL}}} (= \gamma_1 : \gamma_2 : \gamma_3 : \gamma_4)$. Note that here the calculation also includes the relay traffics. If $N_{\text{total}}^{\text{UL}}$ is the total number of uplink slots per frame, the numbers of slots allocated to them are $\frac{\gamma_1 \cdot N_{\text{total}}^{\text{UL}}}{\sum_{i=1}^{4} \gamma_i}$, $\frac{\gamma_2 \cdot N_{\text{total}}^{\text{UL}}}{\sum_{i=1}^{4} \gamma_i}$, $\frac{\gamma_3 \cdot N_{\text{total}}^{\text{UL}}}{\sum_{i=1}^{4} \gamma_i}$, and $\frac{\gamma_4 \cdot N_{\text{total}}^{\text{UL}}}{\sum_{i=1}^{4} \gamma_i}$, respectively. The bandwidth allocation for downlink traffics follows the same way.

However, the above bandwidth allocation is very inefficient because a slot is always allocated to only one SS. In fact, SS 2 and SS 3 can transmit concurrently without interfering

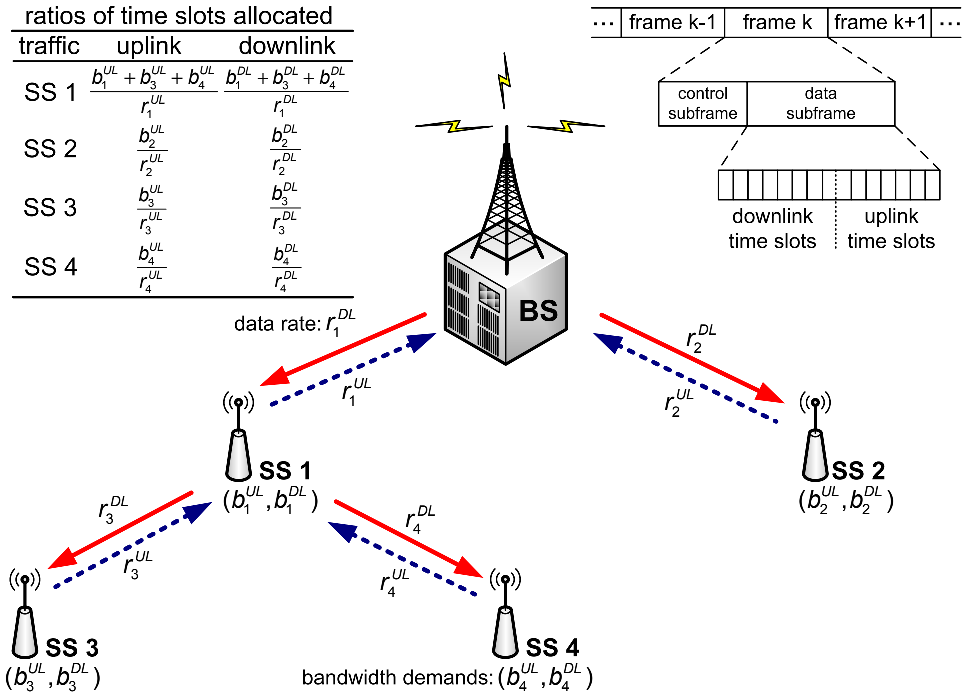

ratios of time slots allocated
traffic
uplink
downlink
SS 1
$\frac{b_1^{UL}+b_3^{UL}+b_4^{UL}}{r_1^{UL}}$
$\frac{b_1^{DL}+b_3^{DL}+b_4^{DL}}{r_1^{DL}}$
SS 2
$\frac{b_2^{UL}}{r_2^{UL}}$
$\frac{b_2^{DL}}{r_2^{DL}}$
SS 3
$\frac{b_3^{UL}}{r_3^{UL}}$
$\frac{b_3^{DL}}{r_3^{DL}}$
SS 4
$\frac{b_4^{UL}}{r_4^{UL}}$
$\frac{b_4^{DL}}{r_4^{DL}}$
… frame k-1 frame k frame k+1 …
control subframe
data subframe
downlink time slots
uplink time slots
BS
data rate: r_1^{DL}
r_1^{UL}
r_2^{DL}
r_2^{UL}
SS 1
(b_1^{UL},b_1^{DL})
SS 2
(b_2^{UL},b_2^{DL})
r_3^{DL}
r_3^{UL}
r_4^{DL}
r_4^{UL}
SS 3
(b_3^{UL},b_3^{DL})
SS 4
bandwidth demands: (b_4^{UL},b_4^{DL})

Figure 2.1: A bandwidth allocation example in the IEEE 802.16 standard.

with each other. We can quantify the waste of slots as follows: Given a routing tree $\mathcal{T}$, the *aggregated uplink bandwidth demand* d_i^{UL} for each SS i is defined as

$$d_i^{\mathrm{UL}} = b_i^{\mathrm{UL}} + \sum_{j \in \mathrm{child}(i)} d_j^{\mathrm{UL}}, \tag{2.1}$$

where child(i) is the set of SS i's children in $\mathcal{T}$. Then, the *demand of uplink transmission time* for SS i is

$$T_i^{\mathrm{UL}} = \frac{d_i^{\mathrm{UL}}}{r_i^{\mathrm{UL}}}. \tag{2.2}$$

Let us denote the sum of uplink transmission time of all SSs by

$$C_{\mathrm{total}}^{\mathrm{UL}} = \sum_{i \in \mathcal{T} - \mathrm{BS}} T_i^{\mathrm{UL}},$$

Therefore, only a ratio of $\frac{T_i^{\mathrm{UL}}}{C_{\mathrm{total}}^{\mathrm{UL}}}$ of the uplink slots are allocated to SS i. However, let the sum of transmission time of SS i and its interference neighbors be

$$C_i^{\mathrm{UL}} = \sum_{j \in E_i} T_j^{\mathrm{UL}}, \tag{2.3}$$

where $E_i = \{i\} \cup \mathcal{I}(i)$ and $\mathcal{I}(i)$ is the set of interference neighbors of SS i. From SS i's perspective, it only sees a ratio of $\frac{C_i^{\mathrm{UL}}}{C_{\mathrm{total}}^{\mathrm{UL}}}$ of the uplink slots to be busy. In other words, the remaining $1 - \frac{C_i^{\mathrm{UL}}}{C_{\mathrm{total}}^{\mathrm{UL}}}$ portion of time is simply idle as seen by SS i. The downlink direction will suffer from the similar waste.

2.2.2 Problem Definition

The problem with the above waste is due to lack of spectral reuse. Our goal is to solve the resource allocation problem in an IEEE 802.16 mesh network with spectral reuse. Given the uplink and downlink bandwidth demands b_i^{UL} and b_i^{DL} and data rates r_i^{UL} and r_i^{DL}, respectively, of each SS i, we will consider the following four issues:

1. Tree reconstruction: How to organize the routing tree according to SSs' bandwidth demands and data rates, so that traffic loads among tree nodes can be balanced and the network throughput can be maximized?

2. Bandwidth allocation: How to allocate time slots to SSs according to their bandwidth demands and data rates, so that SSs can fully utilize the channel?

3. Time-slot assignment: How to assign slots of a frame for SSs with global fairness in mind, so that the transmissions between SSs will not conflict with each other?

4. Bandwidth guarantee: How to schedule real-time and non-real-time traffics when resources are stringent, so that bandwidth requirements of real-time flows can be maintained?

2.3 The Spectral Reuse Framework

In this section, we propose our spectral reuse framework to solve the first three issues in the resource allocation problem. In Section 2.4, we will discuss how to extend our framework to provide bandwidth guarantee for real-time flows. Table 2.2 summarizes the notations used in this work. Fig. 2.2 shows the system architecture of our framework. First, the BS collects the MSH-CSCH:Request messages and passes the bandwidth demands and data rates of SSs to the scheduling and the routing modules. The scheduling module is a fast process, which determines the number of time slots and their positions allocated to each SS in each frame. The routing module is a slow process, which continuously monitors the quality of the routing tree and reconstructs the tree when the quality of the tree degrades. That is, when it is found that the tree cannot efficiently deliver the traffics of SSs, a new routing tree will be computed by the routing module. The BS then broadcasts a MSH-CSCH:Grant message containing the new routing tree and time slot allocation of each SS to the network.

Below, we first present the basic concept of our spectral reuse framework, followed by the designs of the scheduling and the routing modules.

Table 2.2: Summary of notations.

notation	definition
N	number of time slots within a data subframe
$N^{\mathrm{UL}}_{\mathrm{total}}/N^{\mathrm{DL}}_{\mathrm{total}}$	number of uplink/downlink slots within a frame
$N^{\mathrm{UL}}_{i}/N^{\mathrm{DL}}_{i}$	number of uplink/downlink slots allocated to SS i
$b^{\mathrm{UL}}_{i}/b^{\mathrm{DL}}_{i}$	individual bandwidth demand of uplink/downlink traffics generated by SS i
$d^{\mathrm{UL}}_{i}/d^{\mathrm{DL}}_{i}$	aggregated bandwidth demands of uplink/downlink traffics delivered by SS i
$r^{\mathrm{UL}}_{i}/r^{\mathrm{DL}}_{i}$	uplink/downlink data rate of SS i
$T^{\mathrm{UL}}_{i}/T^{\mathrm{DL}}_{i}$	demand of uplink/downlink transmission time of SS i
$\mathcal{I}(i)$	the set of interference neighbors of SS i
E_i	set of SSs that contains SS i and its interference neighborhood $\mathcal{I}(i)$
$C^{\mathrm{UL}}_{i}/C^{\mathrm{DL}}_{i}$	aggregated $T^{\mathrm{UL}}_{j}/T^{\mathrm{DL}}_{j}$ of all SS j in E_i
$C^{\mathrm{UL}}_{\mathrm{total}}/C^{\mathrm{DL}}_{\mathrm{total}}$	aggregated $T^{\mathrm{UL}}_{j}/T^{\mathrm{DL}}_{j}$ of all SS j in the network
$C^{\mathrm{UL}}_{\mathrm{max}}/C^{\mathrm{DL}}_{\mathrm{max}}$	maximal $C^{\mathrm{UL}}_{i}/C^{\mathrm{DL}}_{i}$ among all SS i in the network

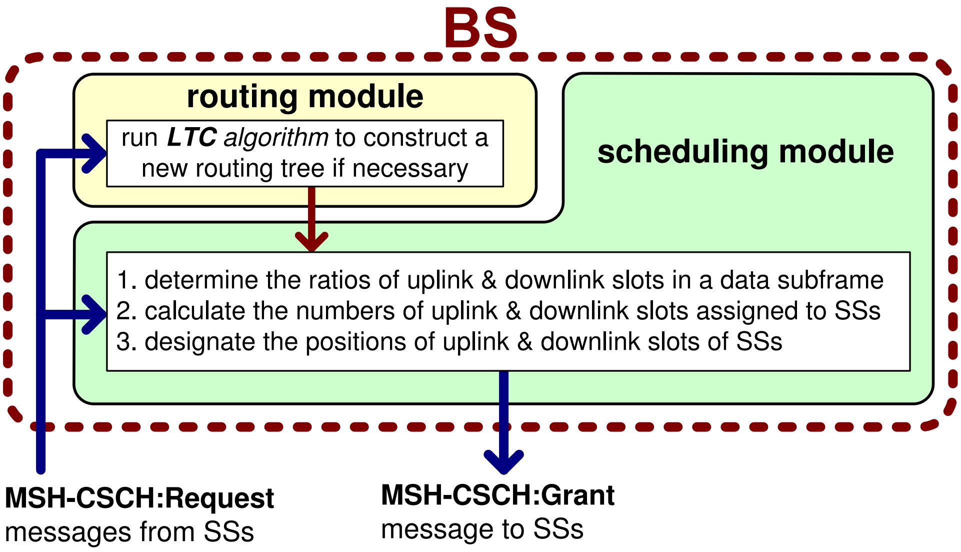

BS
routing module
run *LTC algorithm* to construct a new routing tree if necessary
scheduling module
1. determine the ratios of uplink & downlink slots in a data subframe
2. calculate the numbers of uplink & downlink slots assigned to SSs
3. designate the positions of uplink & downlink slots of SSs
MSH-CSCH:Request messages from SSs
MSH-CSCH:Grant message to SSs

Figure 2.2: System architecture of our spectral reuse framework.

2.3.1 Basic Concept

Earlier, we have indicated that in the uplink case, the scheduling scheme in IEEE 802.16 only assigns $p_i = \frac{T_i^{\mathrm{UL}}}{C_{\mathrm{total}}^{\mathrm{UL}}}$ portion of uplink slots to each SS i. From each SS i's view, the remaining $1 - \frac{C_i^{\mathrm{UL}}}{C_{\mathrm{total}}^{\mathrm{UL}}}$ portion of uplink slots are idle. Ideally, SS i may expect the idle portion to be fairly distributed to all SSs in E_i proportionally. This implies that SS i can share an additional $q_i = \left(1 - \frac{C_i^{\mathrm{UL}}}{C_{\mathrm{total}}^{\mathrm{UL}}}\right) \times \frac{T_i^{\mathrm{UL}}}{C_i^{\mathrm{UL}}}$ portion of uplink transmission time. Thus, the total portion of uplink transmission time assigned to SS i is

$$\frac{T_i^{\mathrm{UL}}}{C_{\mathrm{total}}^{\mathrm{UL}}} + \left(1 - \frac{C_i^{\mathrm{UL}}}{C_{\mathrm{total}}^{\mathrm{UL}}}\right) \times \frac{T_i^{\mathrm{UL}}}{C_i^{\mathrm{UL}}} = \frac{T_i^{\mathrm{UL}}}{C_i^{\mathrm{UL}}}. \tag{2.4}$$

Similarly, the total portion of downlink transmission time assigned to SS i can be upgraded, ideally, to $\frac{T_i^{\mathrm{DL}}}{C_i^{\mathrm{DL}}}$.

Unfortunately, the above Eq. (2.4) does not consider the congestion issue in the global network. In a non-congested network, the uplink bandwidth of an SS should be able to deliver all traffics from itself plus those from its children. Otherwise, congestion on that SS's uplink will occur. Therefore, given a non-congested network, if an SS i's uplink bandwidth is increased by a ratio of α, a sufficient condition to avoid the network becoming congested is to enforce the parent of SS i to increase its uplink bandwidth by at least a ratio of α. Now, let α_i be the ideal ratio of increase by SS i in the uplink direction,

$$\alpha_i = \frac{q_i}{p_i} = \frac{\left(1 - \frac{C_i^{\text{UL}}}{C_{\text{total}}^{\text{UL}}}\right) \times \frac{T_i^{\text{UL}}}{C_i^{\text{UL}}}}{\frac{T_i^{\text{UL}}}{C_{\text{total}}^{\text{UL}}}} = \frac{C_{\text{total}}^{\text{UL}}}{C_i^{\text{UL}}} - 1.$$

The minimum ratio of increase among all SSs is

$$\alpha_{\text{min}} = \min_{\forall i}\{\alpha_i\} = \frac{C_{\text{total}}^{\text{UL}}}{C_{\text{max}}^{\text{UL}}} - 1 \geq 0,$$

where $C_{\text{max}}^{\text{UL}} = \max_{\forall i}\{C_i^{\text{UL}}\}$. Therefore, using α_{min} as the global ratio of increase, the portion of uplink transmission time for each SS i such that the network will not be congested is

$$(1 + \alpha_{\text{min}}) \times \frac{T_i^{\text{UL}}}{C_{\text{total}}^{\text{UL}}} = \frac{T_i^{\text{UL}}}{C_{\text{max}}^{\text{UL}}}.$$

Similarly, the portion of downlink transmission time for each SS i such that the network will not be congested is $\frac{T_i^{\text{DL}}}{C_{\text{max}}^{\text{DL}}}$, where $C_{\text{max}}^{\text{DL}} = \max_{\forall i}\{C_i^{\text{DL}}\}$.

Note that the above calculation includes the demands of individual SSs as well as relay traffics. So our slot allocation is in an end-to-end sense. Next, we discuss how to adopt this concept to the scheduling module to increase channel efficiency. The routing module will reconstruct the routing tree to further improve the performance of the scheduling module. For readability, we first discuss how the scheduling module works, and then present how the routing module works.

2.3.2 Scheduling Module

Given a routing tree $\mathcal{T}$, the scheduling module should properly allocate time slots to SSs in each frame so that the transmissions of nearby SSs will not cause collision and global fairness among SSs can be maintained. Assuming N to be the total number of slots in a data subframe, the scheduling module involves the following steps:

1. We first choose the ratio of the number of uplink slots to the number of downlink slots to be $C^{\mathrm{UL}}_{\max} : C^{\mathrm{DL}}_{\max}$. Thus, the numbers of uplink and downlink slots in a data subframe observed by the BS are $N^{\mathrm{UL}}_{\mathrm{total}} = \left\lfloor \frac{C^{\mathrm{UL}}_{\max}}{C^{\mathrm{UL}}_{\max}+C^{\mathrm{DL}}_{\max}} \times N \right\rfloor$ and $N^{\mathrm{DL}}_{\mathrm{total}} = \left\lfloor \frac{C^{\mathrm{DL}}_{\max}}{C^{\mathrm{UL}}_{\max}+C^{\mathrm{DL}}_{\max}} \times N \right\rfloor$, respectively[1].

2. Based on $N^{\mathrm{UL}}_{\mathrm{total}}$ and $N^{\mathrm{DL}}_{\mathrm{total}}$, we then allocate $N^{\mathrm{UL}}_{i} = \frac{T^{\mathrm{UL}}_{i}}{C^{\mathrm{UL}}_{\max}} \times N^{\mathrm{UL}}_{\mathrm{total}}$ and $N^{\mathrm{DL}}_{i} = \frac{T^{\mathrm{DL}}_{i}}{C^{\mathrm{DL}}_{\max}} \times N^{\mathrm{DL}}_{\mathrm{total}}$ slots to each SS i for its uplink and downlink traffics, respectively. Note that since spectral reuse is considered, it is possible that $\sum_{\forall i} N^{\mathrm{UL}}_{i} > N^{\mathrm{UL}}_{\mathrm{total}}$ and $\sum_{\forall i} N^{\mathrm{DL}}_{i} > N^{\mathrm{DL}}_{\mathrm{total}}$.

3. Next, we need to allocate N^{UL}_{i} collision-free uplink slots in each data subframe to SS i. These slots are divided into two parts. Part 1 contains $\frac{T^{\mathrm{UL}}_{i}}{C^{\mathrm{UL}}_{\mathrm{total}}} \times N^{\mathrm{UL}}_{\mathrm{total}}$ slots. Part 2 contains $\left(\frac{T^{\mathrm{UL}}_{i}}{C^{\mathrm{UL}}_{\max}} - \frac{T^{\mathrm{UL}}_{i}}{C^{\mathrm{UL}}_{\mathrm{total}}}\right) \times N^{\mathrm{UL}}_{\mathrm{total}}$ slots. Part-1 slots are more suitable for real-time traffics because a packet issued by any SS in $\mathcal{T}$ can be delivered to the BS with a latency no more than one frame time (the reason will be explained in Theorem 1). Now we describe how these slots are determined.

   - **Part-1 slots:** These slots are assigned in a bottom-up manner along the tree $\mathcal{T}$. Specifically, we traverse SSs in $\mathcal{T}$ according to the transmission order of MSH-CSCH:Request messages. In IEEE 802.16, such order is reverse in hop-count to the BS (that is, largest hop-count first), and is retained as nodes' IDs in the routing tree for SSs with the same hop-count. Thus, the order of a child SS is always before that of its parent. Following this transmission order, for each SS i being visited, we select the first $\frac{T^{\mathrm{UL}}_{i}}{C^{\mathrm{UL}}_{\mathrm{total}}} \times N^{\mathrm{UL}}_{\mathrm{total}}$ unoccupied slots as its part-1 slots, and then mark these slots as *occupied*. This operation is repeated until all SSs are visited.

[1] Recall that $C^{\mathrm{UL}}_{\max}$ and $C^{\mathrm{DL}}_{\max}$ represent the maximum uplink and downlink demands, respectively, seen by individual nodes. They are bottlenecks of uplink and downlink transmissions. So we use the ratio of $C^{\mathrm{UL}}_{\max}$ and $C^{\mathrm{DL}}_{\max}$ to reflect the demands of uplink and downlink slots and use this ratio to distribute slots. Later on, we will construct the routing tree by minimizing the sum of $C^{\mathrm{UL}}_{\max}$ and $C^{\mathrm{DL}}_{\max}$ to improve spectral reuse. Also, note that the number of slots should be bounded to integers. However, in the following, we will avoid using floor and ceiling functions for ease of presentation.

- **Part-2 slots:** We also assign these slots following the transmission order of MSH-CSCH:Request messages. For every SS i being visited, each of its part-2 slots is selected from the first unoccupied slot by any SS in E_i. Then that slot is marked as occupied. The above operation is repeated until all SSs are visited.

Algorithm 1 gives the pseudo code of the above time-slot assignment scheme.

4. We then designate N_i^{DL} collision-free downlink slots to each SS i. These slots are also divided into two parts, where part 1 contains $\frac{T_i^{\text{DL}}}{C_{\text{total}}^{\text{DL}}} \times N_{\text{total}}^{\text{DL}}$ slots and part 2 contains $\left(\frac{T_i^{\text{DL}}}{C_{\text{max}}^{\text{DL}}} - \frac{T_i^{\text{DL}}}{C_{\text{total}}^{\text{DL}}}\right) \times N_{\text{total}}^{\text{DL}}$ slots. For each part, we assign their slots in a top-down manner along the tree $\mathcal{T}$. Specifically, we traverse SSs in $\mathcal{T}$ by the transmission order of MSH-CSCH:Request messages and then assign slots to these SSs following the reverse order. For each SS being visited, we assign downlink slots to them according to the rules specified in step 3.

Consider an illustrative example in Fig. 2.3, where we need to assign uplink slots for five SSs in the network. Let the demand of each of SSs a, b, c, and d be one slot and the demand of SS e be two slots. We assume that the interference neighborhood of an SS contains all its neighbors within two-hop range. First, part-1 slots can be assigned easily in a sequential manner ($e \rightarrow c \rightarrow d \rightarrow a \rightarrow b$). To assign part-2 slots, observe that the interference neighborhood $\mathcal{I}(a)$ of a includes c, d, and e. For e, we assign slot 8 as its part-2 slot since it is the first unoccupied slot by SSs in $E_e = \{a, c, d, e\}$. Similarly, we assign slot 10 as c's part-2 slot because it is the only unoccupied slot by SSs in $E_c = \{a, b, c, d, e\}$. For a, since $E_a = \{a, c, d, e\}$, we assign slot 9 as its part-2 slot. Note that although slot 9 has already been assigned to b, it does not prevent a from using it because $b \notin E_a$. From Fig. 2.3, we can observe that any packet issued in part-1 slots can always be delivered to the BS within one frame time. However, a packet issued by e in its part-2 slot takes totally 12 slots to be delivered to the BS, which exceeds one frame time. Note that the above scheduling employs a proportional allocation in the sense that the bandwidth allocation for each SS is based on its own bandwidth demand, its children's demands, and the sum of all SSs' demands in the mesh network. The BS collects all SSs' demands and allocates bandwidth to them by the ratio of their aggregated demands and $C^{\mathrm{UL}}_{\mathrm{max}}$. Since all aggregated demands of SSs are divided by the same factor of $C^{\mathrm{UL}}_{\mathrm{max}}$, the resource is proportionally allocated to SSs. Also, once a slot is allocated to an SS, relaying slots are allocated to its parent SS too. Therefore, the allocation is done in an end-to-end perspective.

Algorithm 1: *Time-slot assignment for uplink traffics*

```
Input: numbers of uplink slots for SSs, $\{N^{\mathrm{UL}}_1, \cdots, N^{\mathrm{UL}}_n\}$
Output: result of slot assignment, $transmit[n][N^{\mathrm{UL}}_{\mathrm{total}}]$
// assign part-1 slots
let SS $1, 2, \cdots, n$ be the transmission order of MSH-CSCH:Request messages in $\mathcal{T}$;
$free \leftarrow 1$;
for $i = 1$ to $n$ do
    $allocated \leftarrow free + \frac{T^{\mathrm{UL}}_i}{C^{\mathrm{UL}}_{\mathrm{total}}} \times N^{\mathrm{UL}}_{\mathrm{total}}$;
    for $j = free$ to $allocated$ do $slot[j] \leftarrow i$;
    $free \leftarrow allocated$;

// assign part-2 slots
for $i = 1$ to $n$ do
    for $j = 1$ to $N^{\mathrm{UL}}_{\mathrm{total}}$ do
        $transmit[i][j] \leftarrow NULL$;

for $i = 1$ to $n$ do // mark occupied slots of SSs
    for $j = 1$ to $N^{\mathrm{UL}}_{\mathrm{total}}$ do
        if $slot[j] \in E_i$ then $transmit[i][j] \leftarrow slot[j]$;

for $i = 1$ to $n$ do
    $allocated = \left(\frac{T^{\mathrm{UL}}_i}{C^{\mathrm{UL}}_{\mathrm{max}}} - \frac{T^{\mathrm{UL}}_i}{C^{\mathrm{UL}}_{\mathrm{total}}}\right) \times N^{\mathrm{UL}}_{\mathrm{total}}$;
    for $j = 1$ to $N^{\mathrm{UL}}_{\mathrm{total}}$ do
        if $allocated > 0$ and $transmit[i][j] = NULL$ then
            $transmit[i][j] \leftarrow i$;
            $allocated \leftarrow allocated - 1$;
            for $k = 1$ to $n$ do
                if $k \in E_i$ then $transmit[k][j] \leftarrow i$;
```

Theorem 1 *Part-1 slots are collision-free and any packet issued in part-1 slots can be deliv-*

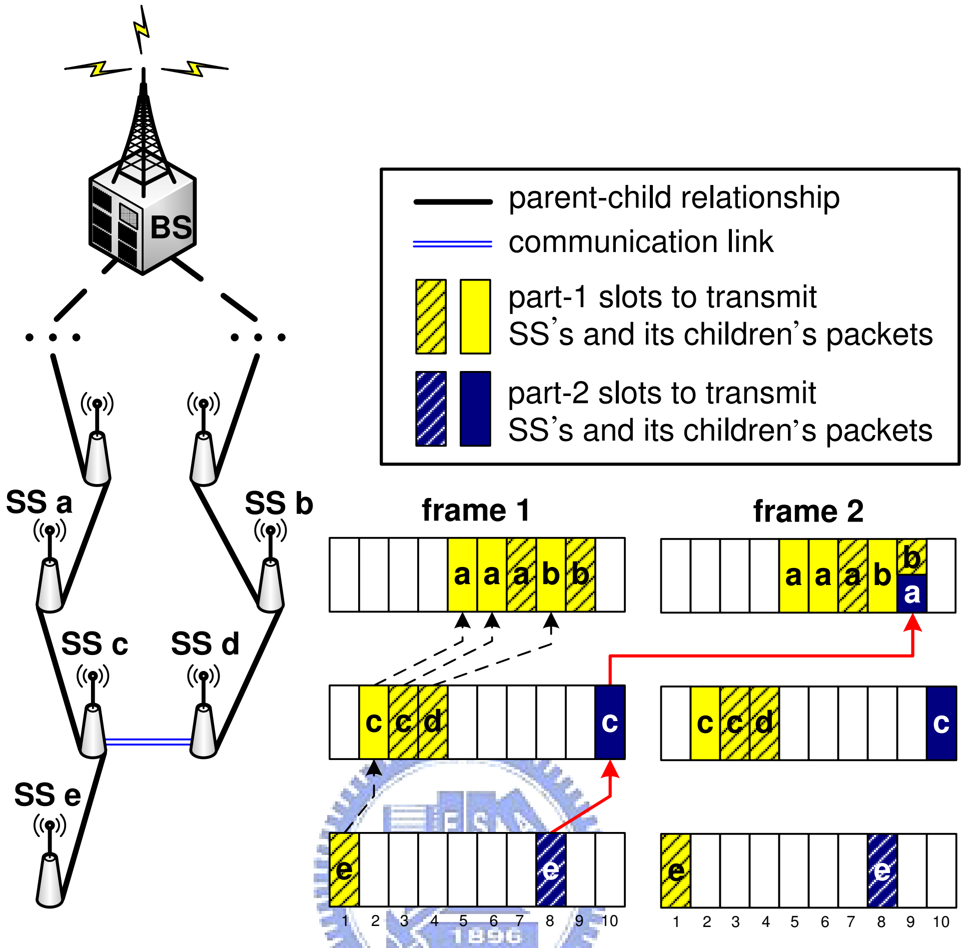

BS
SS a
SS b
SS c
SS d
SS e
parent-child relationship
communication link
part-1 slots to transmit SS's and its children's packets
part-2 slots to transmit SS's and its children's packets
frame 1
frame 2
a a a b b
a a a b b a
c c d c
c c d c
e e
e e
1 2 3 4 5 6 7 8 9 10
1 2 3 4 5 6 7 8 9 10

Figure 2.3: An example of time-slot assignment for uplink traffics.

ered to the destination station within one frame time.

We first prove that part-1 slots are collision-free. For the uplink case, since $\sum_{\forall i} T_i^{\mathrm{UL}} = C_{\mathrm{total}}^{\mathrm{UL}}$, the total number of part-1 slots is $\sum_{\forall i}\left(\frac{T_i^{\mathrm{UL}}}{C_{\mathrm{total}}^{\mathrm{UL}}} \times N_{\mathrm{total}}^{\mathrm{UL}}\right) = N_{\mathrm{total}}^{\mathrm{UL}}$. Thus, there must be enough slots assigned to all SSs for their part-1 slots. In addition, since step 3 in the scheduling module guarantees that any two SSs will not select the same uplink slot, part-1 slots in the uplink case are collision-free. Similarly, for the downlink case, since $\sum_{\forall i}\left(\frac{T_i^{\mathrm{DL}}}{C_{\mathrm{total}}^{\mathrm{DL}}} \times N_{\mathrm{total}}^{\mathrm{DL}}\right) = N_{\mathrm{total}}^{\mathrm{DL}}$, it is guaranteed that there are enough slots assigned to all SSs. Again, since step 4 ensures that two SSs will not choose the same downlink slot, part-1 slots in the downlink case are also collision-free.

We then show that the latency of any packet issued in part-1 slots is bounded to one frame time. For the uplink case, we schedule SSs following the transmission order of MSH-CSCH:Requet messages. Since this order is reverse to the hop-count to the BS, it is guaranteed that we always assign uplink slots of a child SS before its parent. In addition, since each SS has enough uplink slots to relay its children's packets, any packet issued in part-1 slots can be delivered to the BS within one frame time. For the downlink case, since we schedule SSs following the reverse order of the transmission order of MSH-CSCH:Request messages, we will always assign downlink slots of a parent SS before its children. Again, since each SS has enough downlink slots to relay packets from the BS, we can guarantee that any packet from the BS in part-1 slots can be delivered to the destination SS within one frame time.

Theorem 2 *Part-2 slots are collision-free.*

We first prove that part-2 slots in the uplink direction are collision-free. In Section 2.3.1, we have shown that each SS can be assigned with $\frac{T_i^{\mathrm{UL}}}{C_{\mathrm{max}}^{\mathrm{UL}}} \times N_{\mathrm{total}}^{\mathrm{UL}}$ slots without congesting the network. Thus, there are enough slots assigned to all SSs for their part-2 slots. In addition, step 3 in the scheduling module guarantees that any two SSs inside the interference range will not select the same slot. Thus, part-2 slots in the uplink case are collision-free. For the downlink case, since each SS can be assigned with $\frac{T_i^{\mathrm{DL}}}{C_{\mathrm{max}}^{\mathrm{DL}}} \times N_{\mathrm{total}}^{\mathrm{DL}}$ slots without congesting the network, there are also enough slots assigned to all SSs. Similarly, by step 4, we can ensure that two SSs inside the interference range will not choose the same slot. Thus, this theorem still holds in the downlink case.

Remark 1 *The IEEE 802.16 mesh mode only supports time division duplex (TDD) for uplink and downlink traffics. The TDD framing is adaptive in that the bandwidths allocated to uplink and downlink traffics can vary. Unlike the PMP mode, there is no clear boundary between uplink and downlink slots in the mesh mode. In this work, we assume that a slot will be used exclusively by only uplink or downlink throughout the whole network.*

2.3.3 Routing Module

In Section 2.3.1, we have indicated that the uplink and downlink slots allocated to each SS is inversely proportional to the values of $C_{\max}^{\mathrm{UL}}$ and $C_{\max}^{\mathrm{DL}}$, respectively. Therefore, the goal of this routing module is to reconstruct the routing tree, whenever needed, to reduce both $C_{\max}^{\mathrm{UL}}$ and $C_{\max}^{\mathrm{DL}}$ so that SSs can receive more time slots.

Definition 1 *Given a mesh network $\mathcal{G}$, and bandwidth demands and data rates of SSs in $\mathcal{G}$, the* routing tree construction (RTC) problem *is to find a routing tree $\mathcal{T}$ in $\mathcal{G}$ such that the value of $C_{\max}^{\mathrm{UL}} + C_{\max}^{\mathrm{DL}}$ is minimized.*

To prove that the RTC problem is NP-complete, we define a decision problem as follows:

Definition 2 *Given a mesh network $\mathcal{G}$, bandwidth demands and data rates of SSs in $\mathcal{G}$, and a real number $\mathcal{R}$, the* routing tree construction (RTC) problem *is to decide whether $\mathcal{G}$ has a routing tree $\mathcal{T}$ such that $C_{\max}^{\mathrm{UL}} + C_{\max}^{\mathrm{DL}} \leq \mathcal{R}$.*

Theorem 3 *The RTC problem is NP-complete.*

First, given routing trees in $\mathcal{G}$, we can calculate the values of their $C_{\max}^{\mathrm{UL}}$ and $C_{\max}^{\mathrm{DL}}$, and check whether $C_{\max}^{\mathrm{UL}} + C_{\max}^{\mathrm{DL}} \leq \mathcal{R}$. Clearly, this takes polynomial time. Thus, the RTC problem belongs to NP.

We then prove that the RTC problem is NP-hard by reducing a NP-complete problem, the *partition problem* [30], to a special case of the RTC problem in polynomial time. Given a set $\mathcal{X}$ where each element $x_i \in \mathcal{X}$ has an associated size $s(x_i)$, the partition problem asks whether it can partition $\mathcal{X}$ into two subsets with equal total size.

Consider a special case of the RTC problem in Fig. 2.4, where the interference neighborhoods $\mathcal{I}(a)$ and $\mathcal{I}(b)$ of SS a and SS b are not overlapped. The data rates and bandwidth demands of SSs in $E_a \cup E_b$ are set to r and zero, respectively. Except for those SSs in $E_a \cup E_b$, there are n SSs $\mathcal{X} = \{x_1, x_2, \cdots, x_n\}$ connected with both SS c and SS d, each with non-zero equal uplink and downlink bandwidth demands.

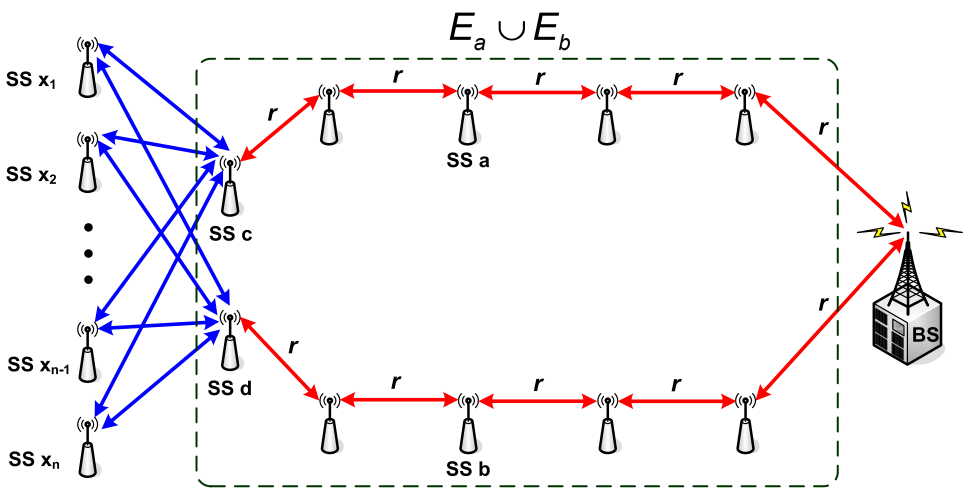


Figure 2.4: A special case of the RTC problem.

Here, we reduce the partition problem to the special case of the RTC problem. Let size $s(x_i)$ be the sum of uplink and downlink bandwidth demands of each $x_i \in \mathcal{X}$, and $\mathcal{R} = \frac{5}{2}\sum_{\forall i}\frac{s(x_i)}{r}$. From Fig. 2.4, we can observe that the parent of $x_i \in \mathcal{X}$ is either SS c or SS d. Because the bandwidth demands of all SSs in $E_a \cup E_b$ are zero, the only way to make $C_{\max}^{\text{UL}} + C_{\max}^{\text{DL}} \leq \mathcal{R}$ is to partition $\mathcal{X}$ into two subsets (where the SSs in $\mathcal{X}$ select either SS c or SS d as their parent) with equal total size. Thus, if there exists a routing tree in $\mathcal{G}$ such that $C_{\max}^{\text{UL}} + C_{\max}^{\text{DL}} \leq \mathcal{R}$, there must be a partition to divide $\mathcal{X}$ into two subsets with equal total size. Obviously, this reduction can be performed in polynomial time. Therefore, the RTC problem is NP-complete.

Below, we propose a heuristic *load-aware tree construction (LTC) algorithm* to deal with the RTC problem. The LTC algorithm constructs the routing tree from leaves to the root. Let $P_i = P_i^{\text{LS}} \cup P_i^{\text{EQ}}$, where P_i^{LS} is the set of SS i's neighbors whose hop counts to the BS are less than that of SS i, and P_i^{EQ} is the set of SS i's neighbors whose hop counts to the BS are equal to that of SS i and these neighbors have already been assigned with parents. The LTC algorithm works as follows:

1. Our goal is to form a routing tree $\mathcal{T}$ to connect all SSs. Initially, SSs are not connecting to any node. So we have a forest of trees, where each tree is an individual SS. Then we can use Eqs. (2.1) and (2.2) to calculate the aggregated uplink bandwidth demand d_i^{UL}, aggregated downlink bandwidth demand d_i^{DL}, demand of uplink transmission time T_i^{UL}, and demand of downlink transmission time T_i^{DL} of each SS i. However, note that to calculate Eq. (2.2), it is necessary to know the parent node of SS i (so as to estimate the transmission rate between i and its parent). To resolve this uncertainty, we assume that before an SS i decides its actual parent, it has a *tentative parent* SS j, where $j \in P_i$ and the transmission rate between i and j is the highest among all candidates.

2. Since the demands of transmission times T_i^{UL} and T_i^{DL} of all nodes i are known, we can apply Eq. (2.3) to calculate C_i^{UL} and C_i^{DL} for all SS i.

3. Let $\mathcal{A}$ be the set of SSs which have not decided their actual parents and which have the maximum hop counts to the BS.

4. This step will decide the actual parent of one SS in $\mathcal{A}$.

   (a) For each SS $i \in \mathcal{A}$, connect SS i to each SS $j \in P_i$ and recompute the new values of C_j^{UL} and C_j^{DL} by assuming that i's actual parent will become j. Note that in order to avoid forming a cycle, if the path from SS i to SS j results in a loop, we set the values of C_j^{UL} and C_j^{DL} as ∞. We then choose the SS j with the minimum value of $C_j^{\mathrm{UL}} + C_j^{\mathrm{DL}}$ as the candidate parent of SS i.

   (b) The above step (a) will choose a candidate parent, say, $p(i)$ for each SS $i \in \mathcal{A}$. Among these candidates, we choose the SS $p(i)$ such that the value of $C_{p(i)}^{\mathrm{UL}} + C_{p(i)}^{\mathrm{DL}}$ is minimized as the actual parent of SS i and make a connection between i and $p(i)$.

5. Repeat step 4, until the set $\mathcal{A}$ is empty.

6. Repeat steps 3, 4, and 5, until all SSs have decided their actual parents.

Step 4(a) is to build the subtree whose subtree root (SS j) has the minimum value of $C_j^{\mathrm{UL}} + C_j^{\mathrm{DL}}$. Similarly, step 4(b) is to build the subtree whose subtree root (SS $p(i)$) has the minimum value of $C_{p(i)}^{\mathrm{UL}} + C_{p(i)}^{\mathrm{DL}}$. This can help balance the distribution of forwarding traffics and keep the final value of $C_{\max}^{\mathrm{UL}} + C_{\max}^{\mathrm{DL}}$ as small as possible in the constructed tree. Note that the above calculations of C_i^{UL} and C_i^{DL} are all tentative. Their values will keep on changing as the tree is building up. Algorithm 2 gives the pseudo code of the LTC algorithm.

Next, we analyze the time complexity of the LTC algorithm. Since each SS has exact one parent, step 4 will be repeated at most n times, where n is the number of SSs in the network. In step 4(a), at most m nodes will be checked and each will check at most d candidates, where m is the maximum number of SSs with the same hop count to the BS and d is the maximum degree of SSs. Thus, the time complexity is $O(nmd)$.

Finally, we comment on the timing to invoke the routing module. Since reconstructing the routing tree causes communication cost, one possible moment to invoke the routing module is when the value of $C_{\max}^{\mathrm{UL}} + C_{\max}^{\mathrm{DL}}$ of the old tree is higher than that of the new tree by a predefined threshold.

2.4 Bandwidth Guarantee for Real-Time Flows

The aforementioned spectral reuse framework can allocate time slots to SSs proportionate to their requests. However, when SSs request new flows or need more bandwidths for their old flows, the system may no longer guarantee enough bandwidths for the original flows. To solve this problem, we propose an *admission control* mechanism to extend our spectral reuse framework. Specifically, we separate flows into *real-time* and *non-real-time* flows. When an SS requests a new flow or more bandwidth for its old flows, we will check whether the

Algorithm 2: *Load-aware tree construction (LTC) algorithm*

```
Input: set $\mathcal{G}$ of all SSs in the network
Output: routing tree $\mathcal{T}$
foreach $i \in \mathcal{G}$ do
    let $r^{\text{UL}}_{j(\max)}$ and $r^{\text{DL}}_{j(\max)}$ be the highest rates of uplinks and downlinks of SS $j$ to SSs in $P_j$;
    $C^{\text{UL}}_i \leftarrow \sum_{j \in E_i} \frac{b^{\text{UL}}_j}{r^{\text{UL}}_{j(\max)}}$;
    $C^{\text{DL}}_i \leftarrow \sum_{j \in E_i} \frac{b^{\text{DL}}_j}{r^{\text{DL}}_{j(\max)}}$;
while $\mathcal{G} \neq \emptyset$ do
    let $\mathcal{A}$ be the set of SSs without parents which have the largest hop counts to the BS;
    $\mathcal{G} \leftarrow \mathcal{G} - \mathcal{A}$;
    while $\mathcal{A} \neq \emptyset$ do
        $C_{\text{min}} \leftarrow \infty$;
        foreach $i \in \mathcal{A}$ do
            foreach $j \in P_i$ do
                calculate $C^{\text{UL}}_j$ and $C^{\text{DL}}_j$ after attaching SS $i$ to SS $j$;
                if $C^{\text{UL}}_j + C^{\text{DL}}_j < C_{\text{min}}$ then
                    $C_{\text{min}} \leftarrow C^{\text{UL}}_j + C^{\text{DL}}_j$;
                    parent $\leftarrow j$;
                    child $\leftarrow i$;
        $\mathcal{T}[\text{child}] = \text{parent}$;
        $\mathcal{A} \leftarrow \mathcal{A} - \{\text{child}\}$;
        foreach $i \in E_{\text{parent}} \cup E_{\text{child}}$ do update $C^{\text{UL}}_i$ and $C^{\text{DL}}_i$;
```

bandwidth requirements of all real-time flows can be still satisfied. If so, we will admit this request. Otherwise, we will reject this request to guarantee bandwidths of existing real-time flows.

Fig. 2.5 illustrates the flowchart of our admission control mechanism. The idea is to prioritize real-time from non-real-time flows. For each SS, we always ensure sufficient slots to satisfy the bandwidth requirements of all its real-time flows, and then distribute the remaining slots to its non-real-time flows. This is what we mean by prioritizing real-time from non-real-time flows. This implies that an SS can always admit more non-real-time flows since its real-time flows always have higher priority. However, when an SS j requests a new real-time flow i (or wants to increases bandwidth of a real-time flow i), the following steps will be

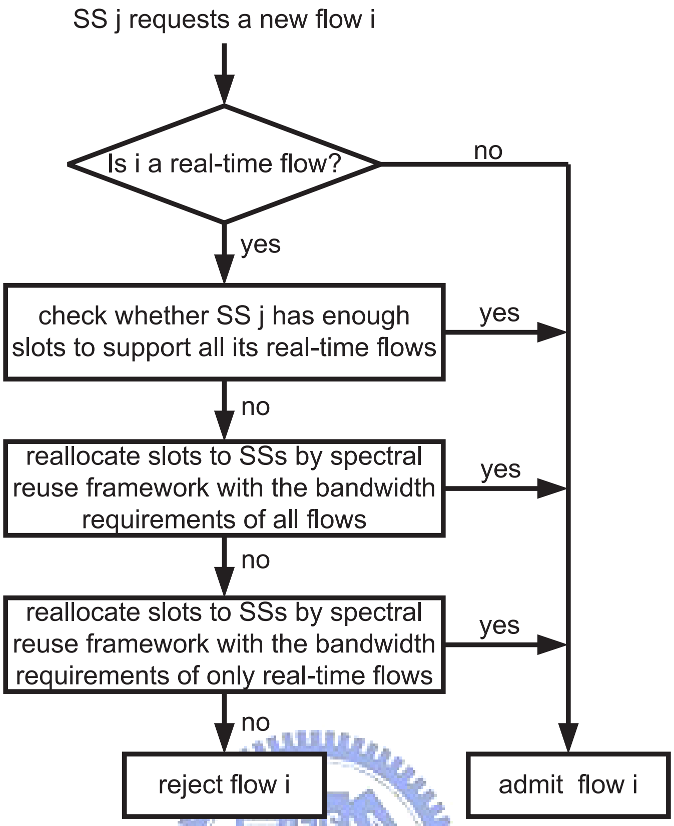

SS j requests a new flow i
Is i a real-time flow?
no
yes
check whether SS j has enough slots to support all its real-time flows
yes
no
reallocate slots to SSs by spectral reuse framework with the bandwidth requirements of all flows
yes
no
reallocate slots to SSs by spectral reuse framework with the bandwidth requirements of only real-time flows
yes
no
reject flow i
admit flow i

Figure 2.5: Flowchart of the admission control mechanism.

executed:

1. Check whether SS j's current slots can support required bandwidths of all its real-time flows (including flow i). If there are enough slots, we can admit flow i. Otherwise, it means that we have to reallocate slots in the system to support this new request (refer to step 2).

2. To reallocate slots of SSs in the network, we will execute our spectral reuse framework in Section 2.3. We will update the bandwidth requirement of SS j, run the routing module to reconstruct the routing tree, and then run the scheduling module to allocate slots to all SSs. Then we check whether this new allocation can support the real-time flows of all SSs. If so, we can admit flow i and adopt the new allocation. Otherwise, it

means that the new scheduling cannot satisfy some real-time flows, so we go to step 3.

3. Update the bandwidth requirements of all SSs by removing their non-real-time flows. With these requirements, we execute our spectral reuse framework again. We run the routing module to reconstruct the routing tree, and then run the scheduling module to allocate slots to all SSs. Then we check whether this new allocation can support the real-time flows of all SSs. If so, we can admit flow i and adopt the new allocation. Otherwise, the system does not have enough slots to support flow i, so we should reject the request of flow i.

Note that although the above step 3 allocates slots to SSs based on their requirements of real-time flows, an SS can still transmit non-real-time flows, as long as its real-time flows do not consume all bandwidths of the SS. Also, we comment that although the above discussions only cover two classes (real-time and non-real-time) of traffics, general multiple m classes of traffics are applicable. In this case, we should check whether the addition of a new flow i (say, in class $k < m$) can still guarantee the bandwidth requirements of all flows in classes $1, 2, \cdots, k$. If not, we can remove flows in classes $k+1, k+2, \cdots, m$ and reallocate slots to check whether the system has enough slots to support the request of flow i.

Next, we formulate our *admission control* mechanism in a mathematical way for implementation. Here we introduce an *uplink channel usage threshold* δ^{UL} to determine whether the network for uplink traffics is congested. Let F_{ctrl} and F_{data} be the ratios of control and data subframes in a frame. According to the scheduling module, we can obtain that

$$\delta^{UL} = \frac{F_{\text{data}}}{F_{\text{ctrl}} + F_{\text{data}}} \times \frac{C^{\text{UL}}_{\text{max}}}{C^{\text{UL}}_{\text{max}} + C^{\text{DL}}_{\text{max}}}.$$

Recall that

$$C^{\text{UL}}_{\text{max}} = \max_{\forall i} \left\{ C^{\text{UL}}_{i} \right\} = \max_{\forall i} \left\{ \sum_{j \in E_i} T^{\text{UL}}_{j} \right\} = \max_{\forall i} \left\{ \sum_{j \in E_i} \frac{d^{\text{UL}}_{j}}{r^{\text{UL}}_{j}} \right\}.$$

Since $C^{\text{UL}}_{\max}$ is the sum of ratios $\frac{d^{\text{UL}}_j}{r^{\text{UL}}_j}$ of the total transmission time allocated to each SS j in E_i of the “bottleneck” SS i, we can use $C^{\text{UL}}_{\max}$ as the degree of uplink channel usage in the network. Specifically, when $C^{\text{UL}}_{\max} \leq \delta^{UL}$, the network for uplink traffics is not congested and thus all uplink flows can receive enough bandwidth to satisfy their QoS requirements. Similarly, we can determine whether the network for downlink traffics is not congested by $C^{\text{DL}}_{\max} \leq \delta^{DL}$, where the *downlink channel usage threshold* δ^{DL} is

$$\delta^{DL} = \frac{F_{\text{data}}}{F_{\text{ctrl}} + F_{\text{data}}} \times \frac{C^{\text{DL}}_{\max}}{C^{\text{UL}}_{\max} + C^{\text{DL}}_{\max}}.$$

Based on the above argument, once $C^{\text{UL}}_{\max} > \delta^{UL}$, the network for uplink direction becomes congested and we have to exclude some flows to alleviate congestion. The idea is to first exclude some non-real-time flows since they do not have stringent deadlines. When the network is still congested even if all non-real-time flows are excluded, we have to exclude some new real-time flows. Here a *new* real-time flow is defined as a real-time flow that does not exist in the previous scheduling result or that changes its bandwidth demand. Given the bandwidth demands of all flows in each SS, the extension of our framework involves the following steps:

1. Run the spectral reuse framework in Section 2.3 to determine the bandwidth allocated to each SS. If $C^{\text{UL}}_{\max} \leq \delta^{UL}$, it means that each SS can obtain enough bandwidth and thus the BS will broadcast the scheduling result to all SSs through the MSH-CSCH:Grant message. Otherwise, we will go to step 2.

2. If there are non-real-time flows in the network, then go to step 3. Otherwise, go to step 5.

3. For each SS i, we check whether its allocated bandwidth can satisfy bandwidth requirements of all its real-time flows. If not, without changing the total bandwidth allocated to SS i, we reduce the bandwidth allocated to SS i’s non-real-time flows until the bandwidth requirements of SS i’s real-time flows can be satisfied. If the bandwidth require-

ments of every SS's real-time flows can be satisfied by the above operation, the BS will broadcast the new scheduling result to all SSs. Otherwise, there must be at least one SS whose allocated bandwidth cannot satisfy its real-time flows even if all its non-real-time flows are excluded. In this case, we will go to step 4.

4. For each SS i, we change its demand of uplink transmission time from T_i^{UL} to $T_{i(\mathrm{RT})}^{UL}$, where $T_{i(\mathrm{RT})}^{UL}$ is the demand of uplink transmission time of all real-time flows in SS i. Then, we execute the scheduling module to recalculate the new result of bandwidth allocated to each SS. After this operation, if $C_{\max}^{\mathrm{UL}} \leq \delta^{UL}$, we will conduct the following actions:

    - If there exists free slots, we will assign them to the non-real-time flows in the network. Specifically, we select the SS i with the minimum value of $T_{i(\mathrm{RT})}^{UL}$ and assign these free slots to its non-real-time flows. We continuously repeat this operation until there is no free slot.

    - Broadcast the new scheduling result to all SSs.

    Otherwise, the network is still congested even if we exclude all non-real-time flows in the network. In this case, we will go to step 5.

5. We then exclude some new real-time flows to alleviate the network congestion. Let $n_{\mathrm{RT\_new}}$ be the total number of new real-time flows in the network. We sort these flows by the following method:

    (a) Select the SS i with the maximum value of $T_{i(\mathrm{RT})}^{UL}$. Then, from SS i, we pick the new real-time flow f_j with the largest bandwidth demand $b(f_j)$.

    (b) Remove f_j from SS i and decrease $T_{i(\mathrm{RT})}^{UL}$ by $\frac{b(f_j)}{r_i}$.

    (c) Repeat the above two steps until all new real-time flows are picked.

We then exclude some new real-time flows from the network by the *binary exclusion*:

(a) Set up two variables $\beta = \frac{1}{2}$ and $k = 2$.

(b) Exclude the first $\lfloor \beta \times n_{\mathrm{RT\_new}} \rfloor$ new real-time flows and execute the spectral reuse framework to check whether $C^{\mathrm{UL}}_{\max} \leq \delta^{\mathrm{UL}}$. If so, it means that we may reject too many new real-time flows. In this case, we update $\beta = \beta - \frac{1}{2^k}$. Otherwise, it means that we have to reject more new real-time flows to alleviate the network congestion. In this case, we update $\beta = \beta + \frac{1}{2^k}$.

(c) Update $k = k + 1$ and repeat step (b) until $n_{\mathrm{RT\_new}} \leq 2^k$.

After the network becomes non-congested by the binary rejection, the BS will broadcast the scheduling result to all SSs.

For the downlink direction, we follow the similar way to exclude some flows to alleviate the network congestion.

2.5 Performance Evaluation

In this section, we present some experimental results conducted by the ns-2 simulator [31] to verify the effectiveness of the proposed framework. We adopt a single-channel OFDM physical layer and a two-ray ground reflection model for radio propagation, and extend the TDMA (time division multiple access) MAC module in ns-2 for the MAC layer. We consider three kinds of network topologies: *regular*, *dense*, and *random*. In a regular network, there are at most 84 SSs placed in a diamond mesh topology, as shown in Fig. 2.6. In a dense network, we add an extra SS in each position marked by '+' in Fig. 2.6. In a random network, we arbitrarily select at most 84 positions from the dense network to place SSs. Note that the resulting network is connected. All SSs are stationary and work in half duplex. The interference neighborhood of an SS includes all its neighbors within two-hop range. So there are at most 12 and 24 nodes in an SS's interference range in the regular and dense networks, respectively. In the random network, an SS's interference range contains 12 nodes in average. There are 512 time slots in a frame. The channel bandwidth is set to 50 Mb/s, and we assume that all links have the same data rates. For each experiment, at least 100 simulations are repeated and we take their average.

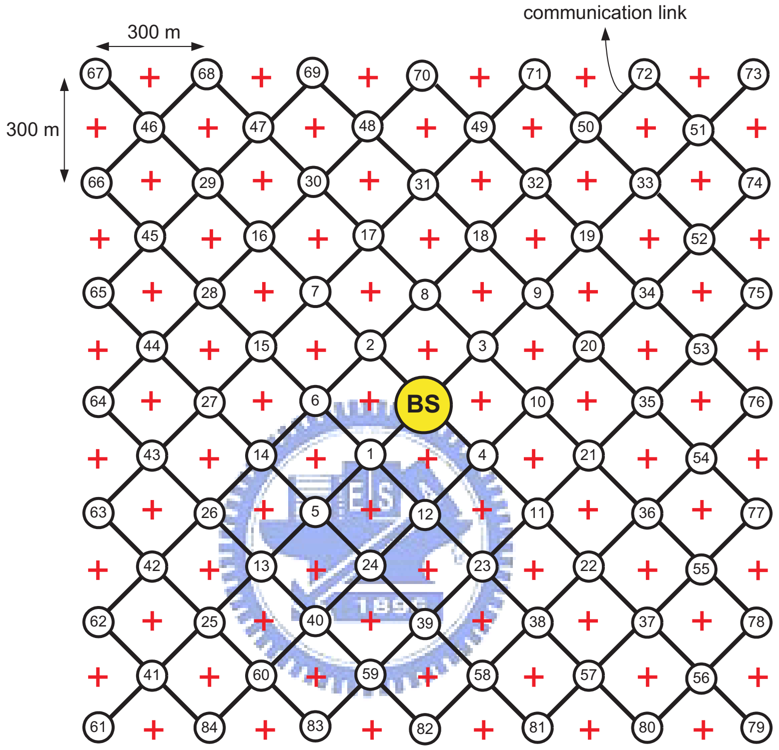


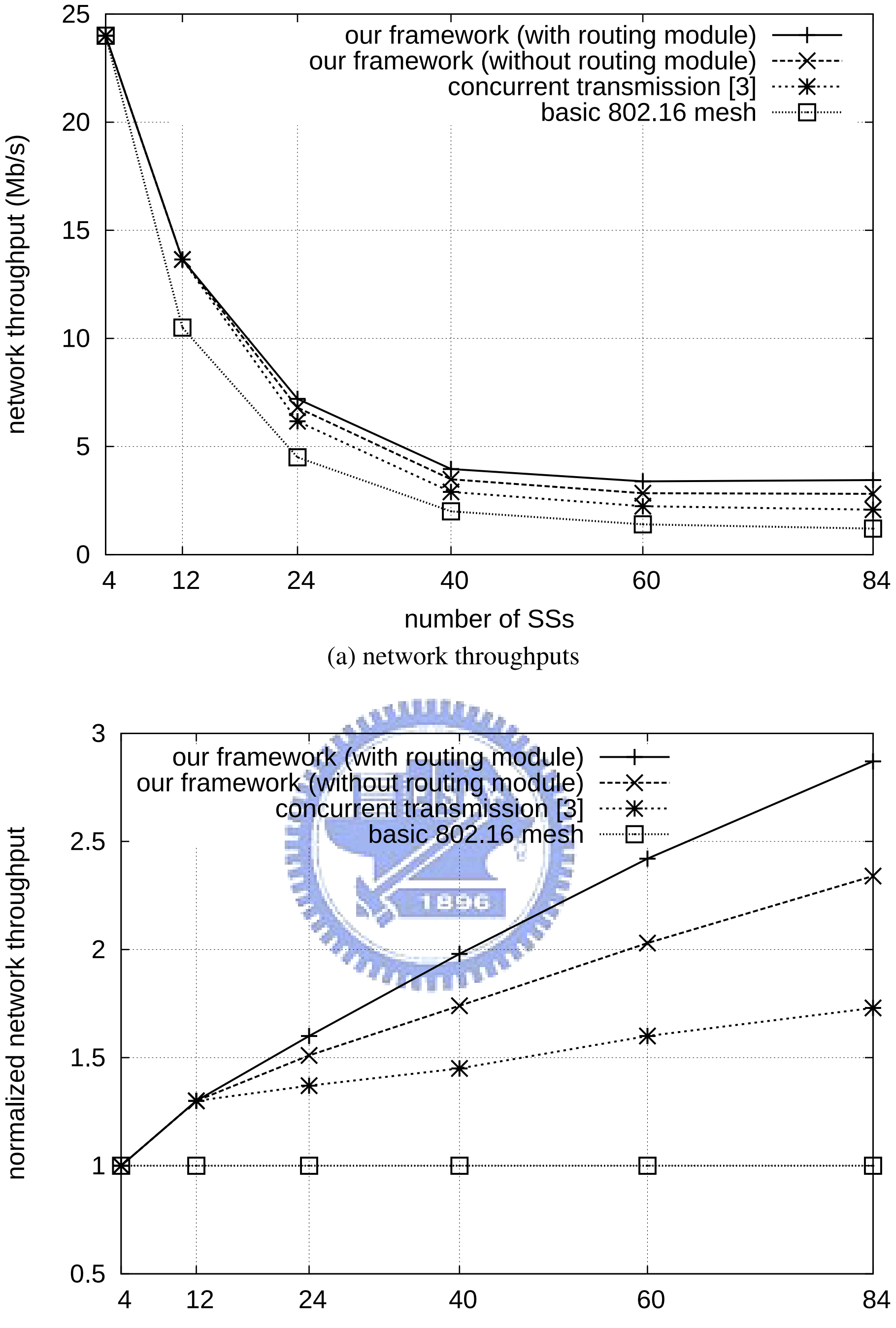
Figure 2.6: The regular and dense network topologies in our experiments.

2.5.1 Network Throughputs under Different Network Topologies

We first evaluate the network throughputs under different network topologies. The *network throughput* is defined as the total amount of data received and transmitted at the BS. We compare our results against the basic 802.16 mesh operation and the concurrent transmission scheme with route adjustment proposed in [3]. For the 802.16 operation, the random routing tree is adopted and the numbers of uplink and downlink slots are set to equal. Each SS will generate random traffic loads and request the same uplink and downlink bandwidth demands. For the regular and random networks, the number of SSs is set to 4, 12, 24, 40, 60, and 84. For the dense network, we set the number of SSs as 8, 24, 48, 80, 120, and 168.

Fig. 2.7 shows the network throughputs of different methods in the regular network. Clearly, the network throughput will decrease as the number of SSs increases because a packet needs to travel more hops in average as the network scales up. From Fig. 2.7, we can observe that the throughput of the 802.16 operation drops significantly when the number of SSs increases. This is because it adopts a random routing tree, which causes longer relay routes. Moreover, the neglect of spectral reuse greatly hurts the system performance. The improvement of throughput by the concurrent transmission scheme proposed in [3] is limited because it constructs the routing tree according to the SSs' positions, rather than their traffic loads. Thus, the network bottleneck cannot be reflected and the benefit of route adjustment is limited. Besides, this concurrent transmission scheme restricts that SSs cannot transmit data earlier than their child SSs so that the throughput is reduced. Our framework performs better than these two schemes because it can estimate the degree of spectral reuse according to SSs' traffic loads and thus allocates more time slots to SSs. As the network scale grows, the degree of spectral reuse can also increase. In addition, the LTC algorithm of the tree module can generate better routing paths to distribute the traffics more evenly. Therefore, the complete framework can result in the highest throughput.

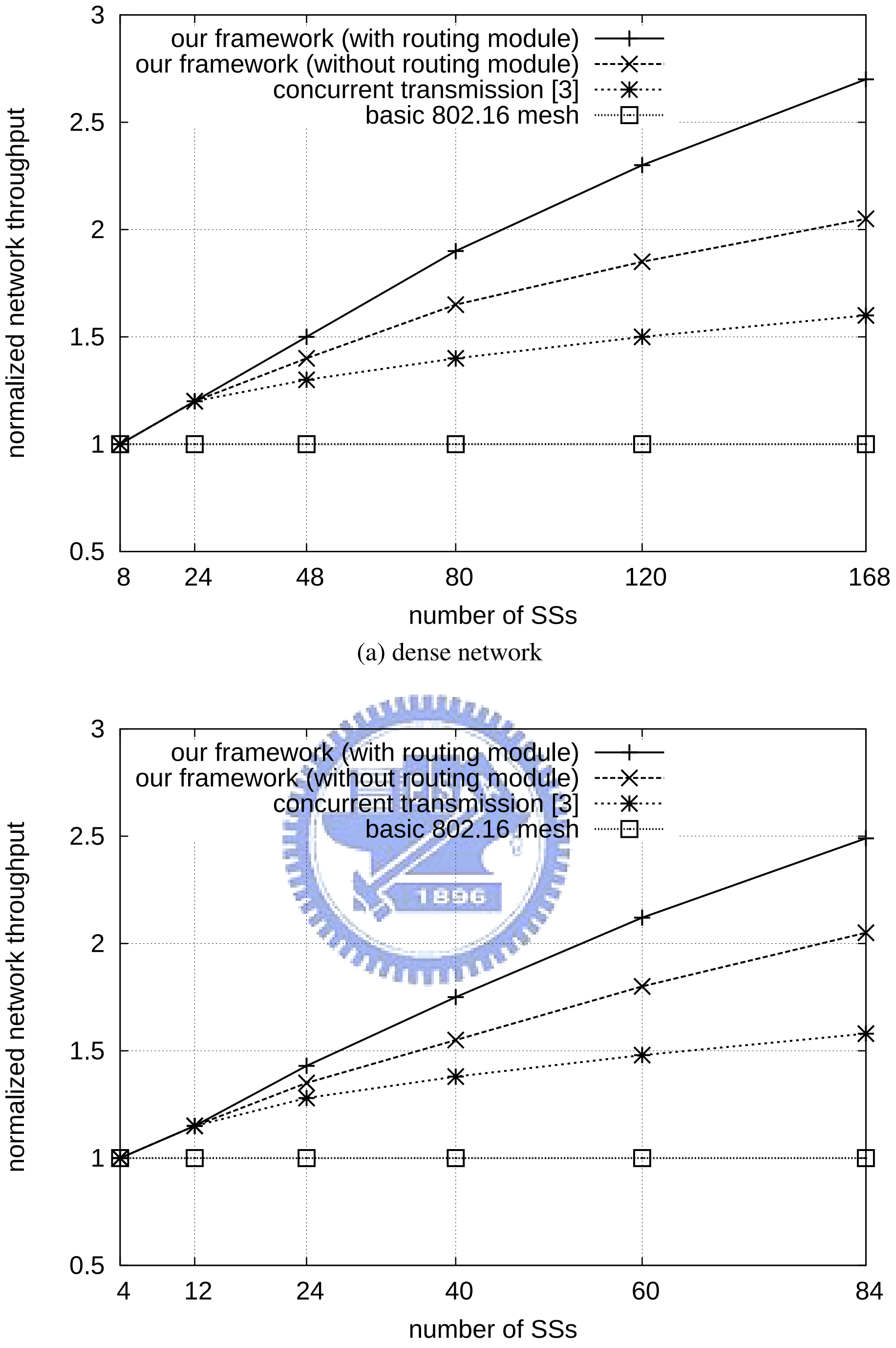
We then verify the network throughputs of different methods in the dense and random networks, as shown in Fig. 2.8. All network throughputs are normalized by that of the basic 802.16 mesh operation. From Fig. 2.8, we can observe that the results are similar to that in Fig. 2.7. However, as compared with Fig. 2.7, the improvement of our framework slightly degrades. For the dense network, this is due to the decrease of degree of spectral reuse since the number of nodes in each SS's interference neighborhood becomes double. For the random network, this is because the network bottleneck usually appears in the one-hop neighbors of



(a) network throughputs

(b) normalized network throughputs

Figure 2.7: Comparison of network throughputs in the regular network.



(a) dense network

(b) random network

Figure 2.8: Comparison of normalized network throughputs in the dense and random networks.

the BS.

In the following experiments, we conduct all simulations in the regular network.

2.5.2 Network Throughputs under Different Traffics Demands

Fig. 2.9 shows the normalized network throughputs under different number of SSs with various uplink traffic demands. Each SS randomly requests 50% to 100% uplink bandwidth demand. From Fig. 2.9, we can observe that the network throughput of our framework is much higher than that of the 802.16 operation. This is because the 802.16 operation only allocates equal numbers of slots to uplink and downlink traffics without any flexibility. The situation becomes worse when the number of SSs increases, because the difference between the amount of uplink traffics and the amount of downlink traffics could be large. On the contrary, our framework allocates the ratio of uplink to downlink slots as $C_{\max}^{\mathrm{UL}} : C_{\max}^{\mathrm{DL}}$, which reflects the practical traffic loads of SSs. In addition, the tree module helps reconstruct a better routing tree to reduce both the values of $C_{\max}^{\mathrm{UL}}$ and $C_{\max}^{\mathrm{DL}}$, thereby further improving the system performance.

Fig. 2.10 illustrates the normalized network throughputs under different uplink traffic demands. We set the number of SSs as 84. Each SS generates 0.3 Mb/s traffic load in average, where the ratio of uplink request is varied from 10% to 50%. From Fig. 2.10, we can observe that our framework can significantly improve the network throughput, especially when the difference between uplink and downlink traffic demands increases. This is because the 802.16 operation simply allocates equal numbers of slots for uplink and downlink traffics, which may lead to network congestion in one direction while leave slots wasted in another direction. The situation becomes worse when the traffic loads in uplink and downlink directions become extremely unbalanced.

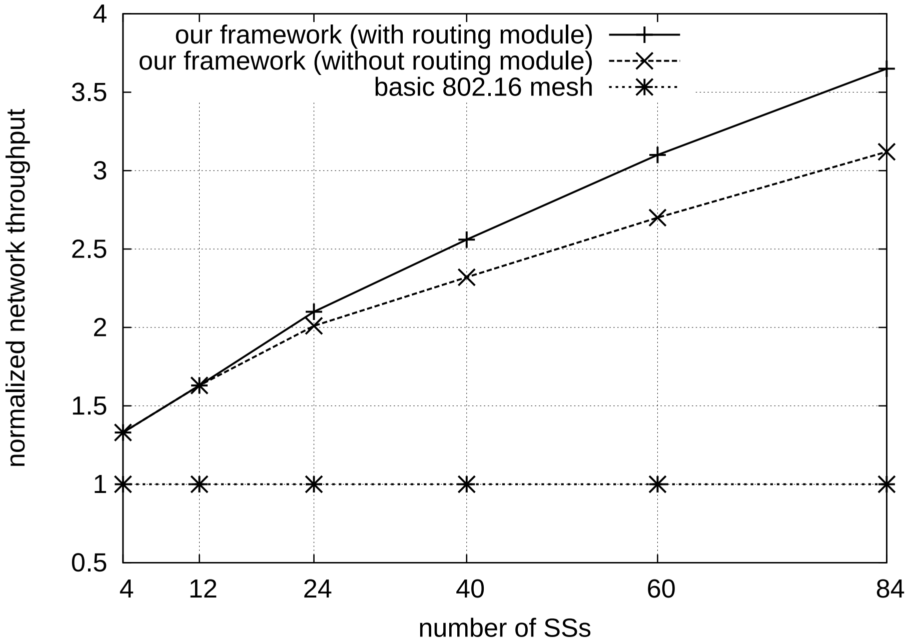


Figure 2.9: Comparison of normalized network throughputs under different number of SSs with various uplink traffic demands.

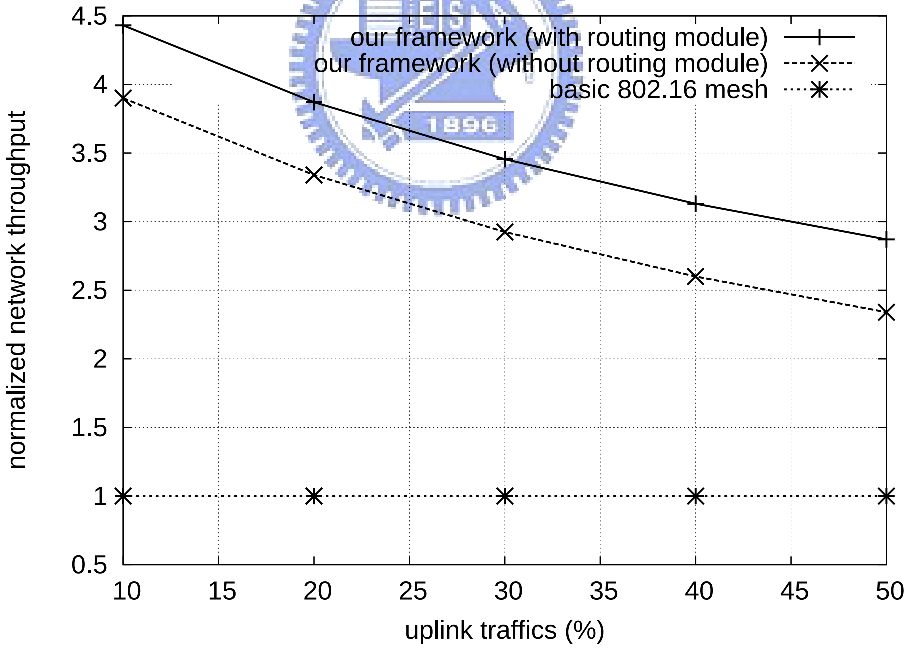


Figure 2.10: Comparison of normalized network throughputs under different uplink traffic demands.

2.5.3 Packet Dropping Ratio of Real-Time Flows

We then evaluate the *packet dropping ratio* of real-time flows in the network, which is defined as the ratio of the number of real-time packets dropped (due to exceeding deadlines) to the number of real-time packets generated. We set the deadline of a real-time packet as 500 ms. There are 80% real-time flows and 20% non-real-time flows in the network. Fig. 2.11 illustrates the packet dropping ratios under different number of SSs. We can observe that our framework can result in a lower packet dropping ratio because it can achieve a higher network throughput with the help of spectral reuse and tree reconstruction. Therefore, real-time flows can receive more bandwidths to alleviate their packet dropping ratios.

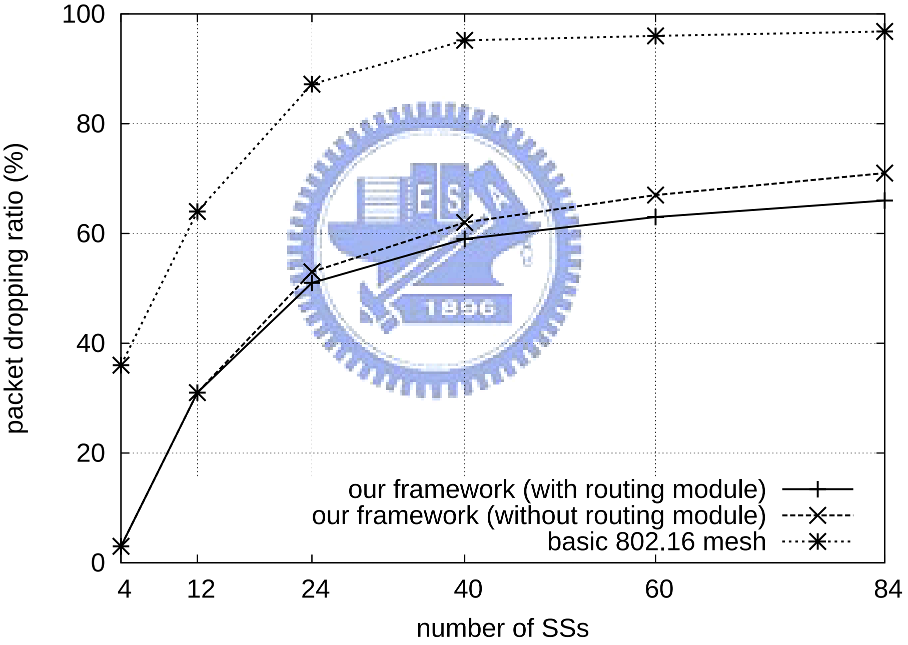


Figure 2.11: Comparison of packet dropping ratios under different number of SSs.

2.5.4 Real-Time Flow Granted Ratio

Fig. 2.12 shows the real-time flow granted ratio under different number of SSs. The *real-time flow granted ratio* is defined as the ratio of the number of admitted real-time flows to the number of requested real-time flows. We set the ratio of the number of real-time flows to the number of non-real-time flows as $4 : 1$. Each flow uniformly generates a traffic load of [0.1 Mb/s, 0.5 Mb/s]. From Fig. 2.12, we can observe that when the number of SSs increases, the real-time flow granted ratio will decrease because the average routing path to the BS increases. In this case, SSs have to relay more traffics from their children, resulting in a high risk of network congestion. By exploiting spectral reuse, our framework can achieve a higher network throughput and thus improves the real-time flow granted ratio. Besides, the extension of our framework in Section 2.4 prioritizes real-time from non-real-time flows, thereby further improving the real-time flow granted ratio.

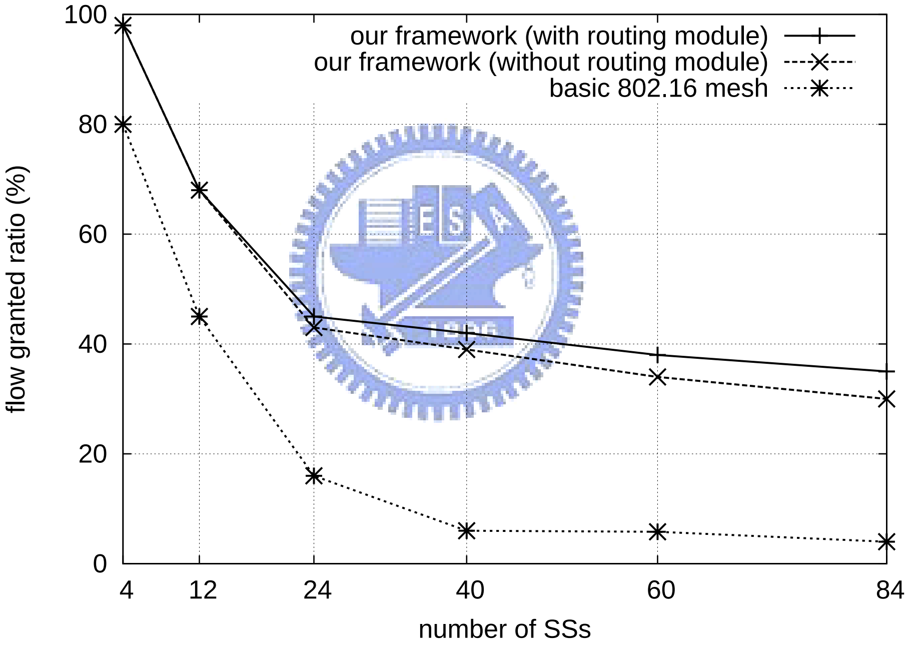


Figure 2.12: Comparison of real-time flow granted ratios under different number of SSs.

Fig. 2.13 illustrates the real-time flow granted ratio under different traffic loads of 84 SSs. We vary the average traffic load of SSs from 0.1 Mb/s to 0.6 Mb/s. Each SS will request 80% real-time flows and 20% non-real-time flows. From Fig. 2.13, we can observe that the real-time flow granted ratio decreases significantly as the average traffic load increases due to the serious network congestion. In such a severe environment, the 802.16 operation can only admit no more than 10% real-time flows. On the other hand, our framework can still admit 25% real-time flows even when the average traffic load of SSs arrives to 0.6 Mb/s. This reflects the flexibility of the flow scheduling in our framework.

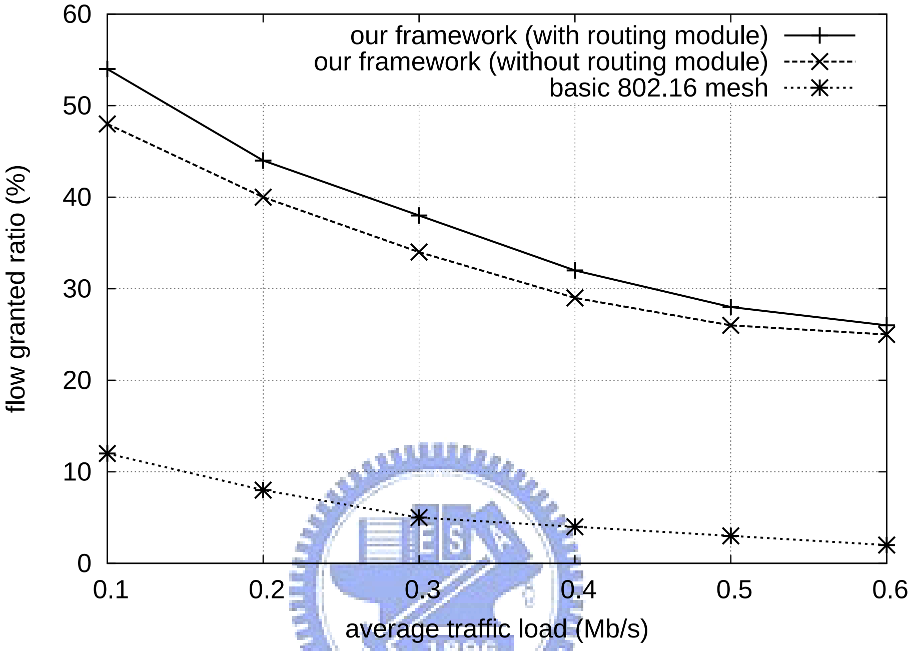


Figure 2.13: Comparison of real-time flow granted ratios under different traffic loads.

Fig. 2.14 shows the real-time flow granted ratio under different non-real-time traffic demands. We set the number of SSs as 84. Each SS generates 0.3 Mb/s traffic load in average, where the ratio of non-real-time flows is varied from 10% to 50%. From Fig. 2.14, we can observe that the real-time flow granted ratio of our framework can be improved as the ratio of non-real-time flows increases because real-time flows can obtain more bandwidths from these non-real-time flows.

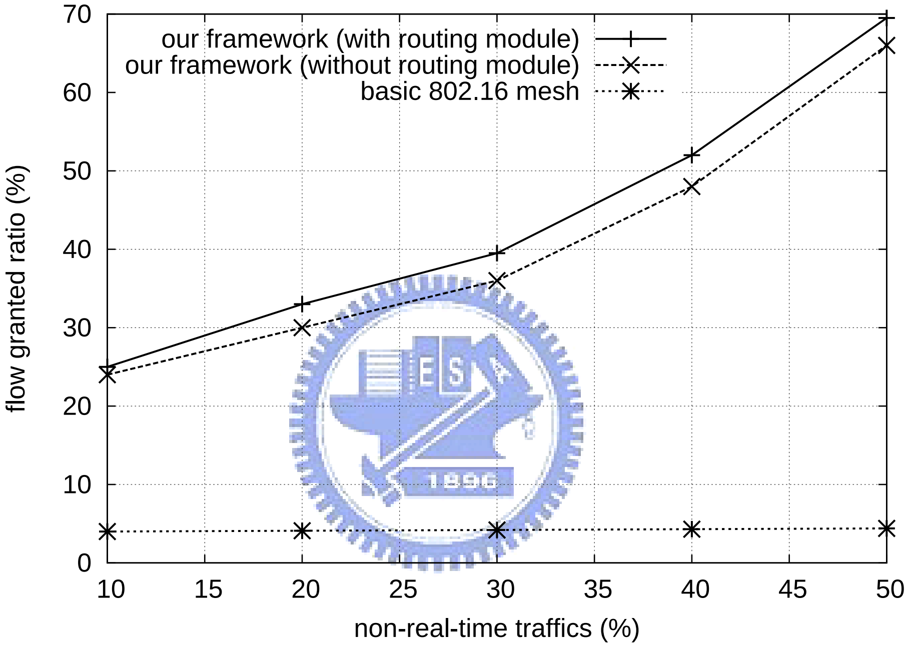


Figure 2.14: Comparison of real-time flow granted ratios under different non-real-time traffic demands.

Chapter 3

Route Throughput Analysis with Spectral Reuse for IEEE 802.16 Relay Networks

3.1 Observations and Motivations

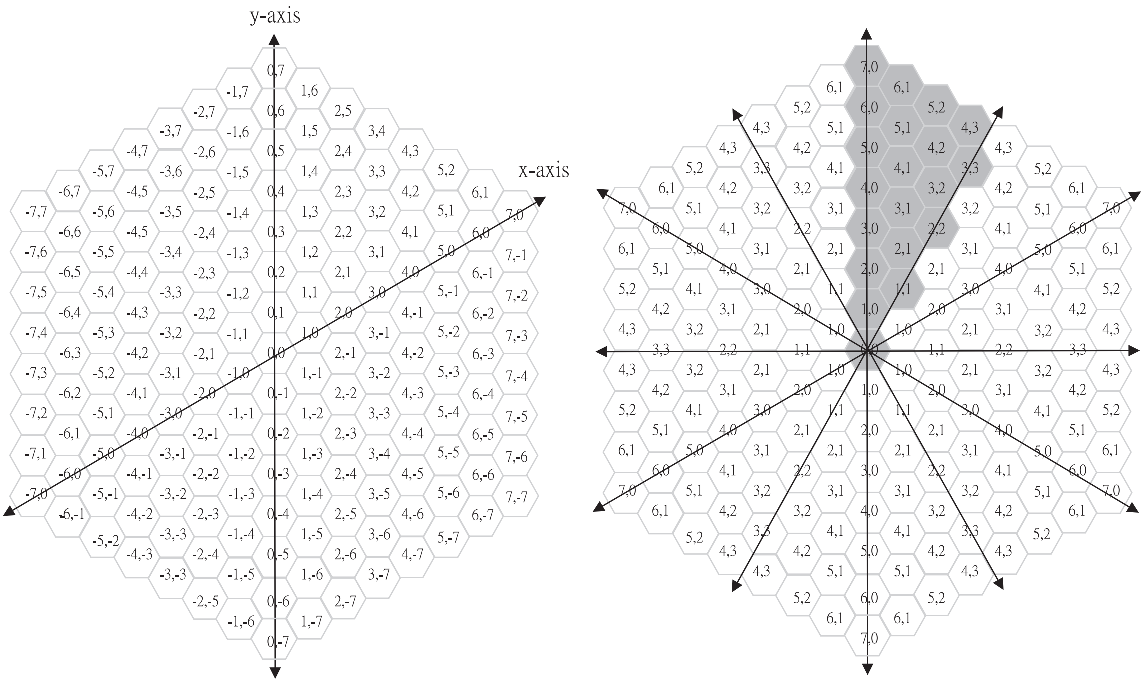
In this work, we consider a mobile ad hoc network (MANET) consisted of mobile relay stations where each wireless link can support multiple rates and has the auto-rate selection capability. Given a routing path, this work provides an analytic tool to evaluate the expected throughput of the route with spectral reuse, assuming that hosts move following the discrete-time, random-walk model. The result can be added as a new metric for route selection in MANETs. (We comment that we do not intend to propose a new routing protocol here. But the proposed results may be used in many current protocols to compute a new route selection metric.)

The MANET is a flexible and dynamic architecture that is attractive due to its ease in network deployment. Routing is perhaps one of the most intensively addressed issues in MANET. Many different criteria have been used in route selection, including hop count [8], signal strength [9], route lifetime [10], and energy constraint [11]. Among these metrics, hop count may be the most widely used metric in choosing routes. When a hop-count based routing protocol is given multiple paths, the shortest path is normally selected and a random path is selected when there is a tie. This metric has the advantage of simplicity, requiring no additional measurements and incurring the least number of transmissions. The primary disadvantage of this metric is that it does not take packet loss or available bandwidth into account, especially when network interfaces can transmit at multiple rates [12]. It has been shown in [13] that a route which minimizes the hop count does not necessarily maximize the throughput of a flow.

While it is true that there is no single route selection metric that is able to best fit all possible routing scenarios in MANET, few works have considered the impact of multi-rate communication capability that is widely supported by many current wireless LAN products. For example, IEEE 802.11b supports rates of 11, 5.5, 2, and 1 Mbps, while IEEE 802.11a supports rates of 6, 9, 12, 18, $\cdots$, and 54 Mbps. Route selection is more complicated in a multi-rate MANET than in a single-rate environment. Also, there exists an inherent tradeoff between transmission rates and their effective transmission ranges [14]. To support reliable data transmissions, longer-range communications must use lower rates, and vice versa. Auto-rate selection protocols [15, 16] do exist at the link level. Reference [17] proposes a multi-rate-aware topology control algorithm to enhance the network throughput in multi-hop ad hoc networks, and [18] uses fast links (with high nominal bit rate) to improve the system throughput in wireless mesh networks. However, they only focus on static network environment without taking mobility into account. Reference [19] proposes a multi-rate-aware sub-layer between the MAC and the network layers to improve resource utilization and to minimize power consumption, but the effect of multi-rate communications at the routing level is not yet fully addressed.

3.2 System Model

In this work, we assume that each mobile host roams around in the network area following the discrete-time, random-walk mobility model, which has been widely used in several works [32–34]. In this model, the network area is partitioned into a number of hexagonal cells, each



(a)

(b)

Figure 3.1: (a) a cellular system to model station mobility, and (b) the “folding” of link states.

with radius r and each with a coordinate (x, y), as shown in Fig. 3.1(a). Cells on the x-axis are numbered $(x, 0)$, and those on the y-axis $(0, y)$. The coordinates of other cells are obtained by mapping them onto these two axes, as is normally done in the Cartesian coordinate system.

Although hosts actually roam around in continuous time domain, we will work in discrete time domain by dividing time into fixed-length units. We assume that mobile hosts roam around in a cell-to-cell basis following the random walk model. Given a mobile host at any cell, it will move into any one of its six neighboring cells in the next time unit with an equal probability of $1/6$.

Cells in the network are further divided into layers as follows. Cell $(0, 0)$ is on the layer-0. The six neighboring cells of cell $(0, 0)$ are the layer-1 cells, and the outer cells surrounding layer-i cells are said to be on layer $(i+1)$. The number of cells included in an n-layer network is $3n^2 + 3n + 1$. In this work, we will model the transmission range of a mobile host by a certain number of layers, by assuming its current location at layer 0.

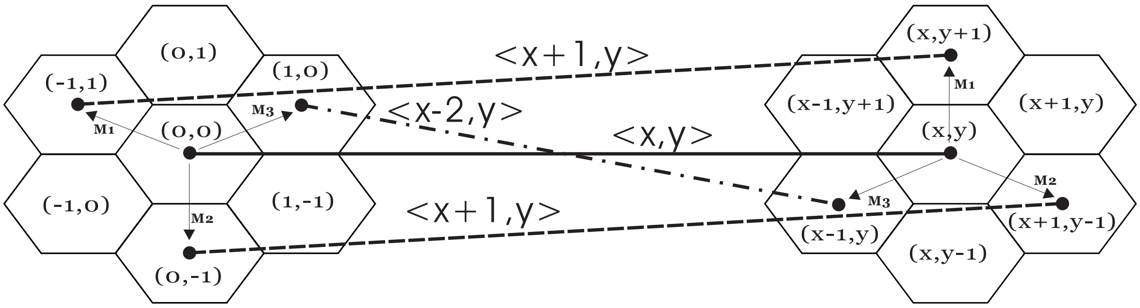


Figure 3.2: Example of link state changes.

Table 3.1: The probability distribution for a wireless link to switch from state $\langle x, y\rangle$ to state $\langle x', y'\rangle$ after one time unit.

<x',y'>	<x,y>	<x-1,y>	<x-1,y-1>	<x,y-2>	<x+1,y-2>	<x+1,y-1>	<x+1,y>	<x,y-1>	<x+2,y-2>	<x+2,y-1>
Probability	6/36	2/36	2/36	1/36	2/36	2/36	2/36	2/36	1/36	2/36

<x',y'>	<x+1,y+1>	<x,y+1>	<x+2,y>	<x,y+2>	<x-1,y+2>	<x-1,y+1>	<x-2,y+2>	<x-2,y+1>	<x-2,y>
Probability	2/36	2/36	1/36	1/36	2/36	2/36	1/36	2/36	1/36

Following [35], we use a vector to represent the state of a wireless link. Specifically, given a wireless link between two hosts located at cells (x, y) and (x', y'), we represent the link's state as a vector $\langle x' - x, y' - y\rangle$. A routing path thus may contain a sequence of vectors, each representing a wireless link. For example, a routing path containing hosts in cells $(0, 1)$, $(3, 1)$, and $(7, -3)$ in the order can be written as $[\langle 3, 0\rangle, \langle 4, -4\rangle]$.

Based on the random walk model, we can derive a probability model for the state change of a wireless link. Let $\langle x, y\rangle$ be the state of a wireless link connecting two neighboring hosts at time t, At time $t + 1$, each of the hosts may move into one of its six neighboring cells with probability of $1/6$. This gives 36 combinations of the two hosts' next locations (as shown in Fig. 3.2), which can be reduced to 19 link states with different probabilities (as shown in Table 3.1) [35].

Suppose that the transmission distance of a host is n layers. Then the number of states for a wireless link will be as large as $3n^2 + 3n + 1$. To prevent the problem of state explosion that so many states need to be taken into consideration, [36] proposes to merge equivalent cells by

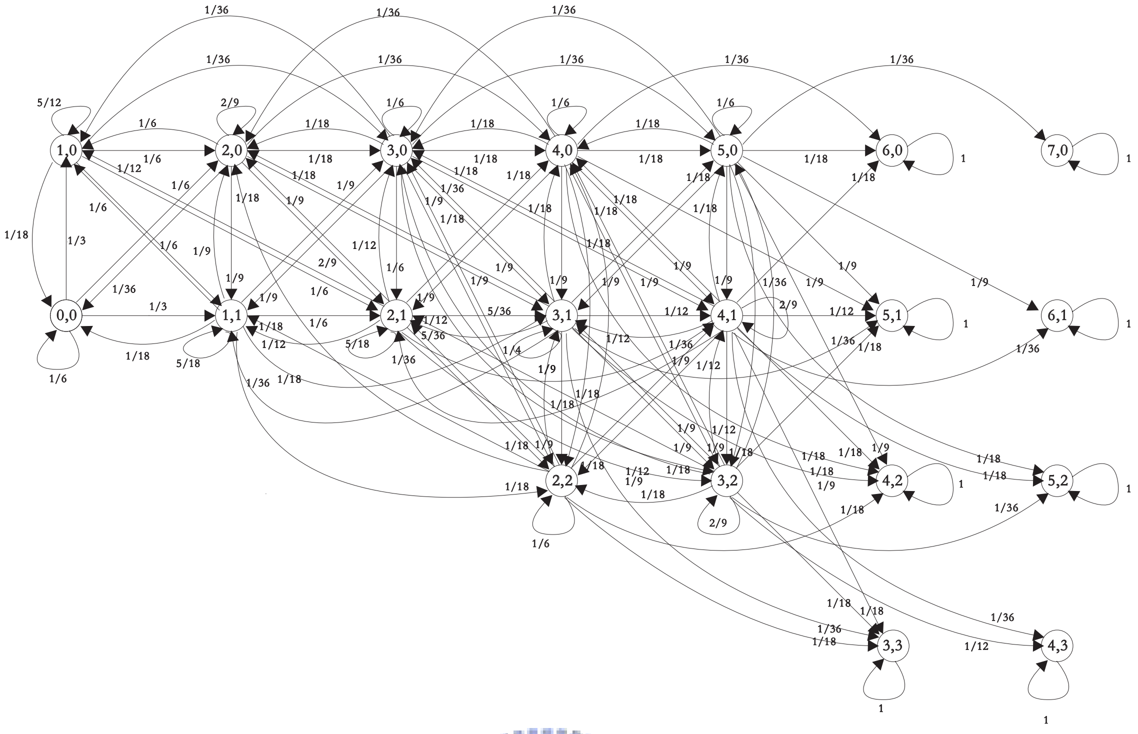


Figure 3.3: State transition diagram of a wireless link when $n = 5$.

"folding" the 12 sectors in Fig. 3.1(b) into one (cells of the same indices are equivalent). This reduces the number of states by around 1/12. Detailed derivations can be found in [36]. The basic observation is that all cells at the same distance from the central cell along the 30-, 90-, 150-, 210-, 270, and 330-degree axes will have the same probability distribution. This divides the cellular network into 6 equivalent sectors. Further moving from boundaries of each sector toward its central part, we see, based on symmetry property, that the sector can be folded by half, thus further reducing the number of states by half. Depending of the value of n, the new number of states becomes:

$$C(n) = \begin{cases} 1 & n = 0 \\ \frac{(n+1)(n+3)}{4} & n > 0 \text{ and n is odd} \\ \frac{n(n+4)}{4} + 1 & n > 0 \text{ and n is even} \end{cases} .$$

Most current wireless LAN cards support automatic rate selection depending on channel conditions. For example, IEEE 802.11b standard supports four transmission rates: 11, 5.5,

$$M = \begin{array}{c|ccccccc|}
 & \langle 0,0\rangle & \langle 1,0\rangle & \langle 2,0\rangle & \langle 1,1\rangle & \cdots & \langle 5,2\rangle & \langle 4,3\rangle \\
\hline
\langle 0,0\rangle & \frac{6}{36} & \frac{12}{36} & \frac{6}{36} & \frac{12}{36} & \cdots & \frac{0}{36} & \frac{0}{36} \\
\langle 1,0\rangle & \frac{2}{36} & \frac{15}{36} & \frac{6}{36} & \frac{6}{36} & \cdots & \frac{0}{36} & \frac{0}{36} \\
\langle 2,0\rangle & \frac{1}{36} & \frac{6}{36} & \frac{8}{36} & \frac{4}{36} & \cdots & \frac{0}{36} & \frac{0}{36} \\
\langle 1,1\rangle & \frac{2}{36} & \frac{6}{36} & \frac{4}{36} & \frac{10}{36} & \cdots & \frac{0}{36} & \frac{0}{36} \\
\vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\
\langle 5,2\rangle & \frac{0}{36} & \frac{0}{36} & \frac{0}{36} & \frac{0}{36} & \cdots & \frac{36}{36} & \frac{0}{36} \\
\langle 4,3\rangle & \frac{0}{36} & \frac{0}{36} & \frac{0}{36} & \frac{0}{36} & \cdots & \frac{0}{36} & \frac{36}{36}
\end{array}$$

Figure 3.4: A state transition matrix of a wireless link when $n = 5$.

2, and 1 Mbps. When the MAC layer overheads are taken into account (control overheads, contention overheads, collision costs, etc.), the effective link rates may be reduced from 11, 5.5, 2, and 1 Mbps to 4.55, 3.17, 1.54, and 0.85 Mbps, respectively [14]. We assume that the rate of a wireless link will depend on the distance between the two hosts of the link. Reference [14] provides a general theoretical model of the attainable throughput in multi-rate ad hoc wireless networks.

3.3 Route Throughput Analysis

A route consists of a number of wireless links. Given a routing path, our goal is to determine the expected route throughput based on the random walk model. In the previous section, we have derived how a wireless link changes states. Suppose that each mobile host has a transmission range of n layers. Then we can model a wireless link by considering an $(n+2)$-layer network. For example, Fig. 3.3 shows the state transition diagram of a wireless link when $n = 5$. Note that states $\langle 6,0\rangle, \langle 5,1\rangle, \langle 4,2\rangle, \langle 3,3\rangle, \langle 7,0\rangle, \langle 6,1\rangle, \langle 5,2\rangle$, and $\langle 4,3\rangle$ are "absorbing" states such that $x + y > n$ for state $\langle x, y\rangle$, which means the distance between mobile hosts is larger than the transmission range and once a wireless link changes to any of these states, the link is considered broken.

The state transition probability of a wireless link in Fig. 3.3 can be modeled by a matrix M in Fig. 3.4, where each element $M_{i,j}$ represents the probability for a link to transit from state i to state j. M^k is the k-th power of M, which represents the state transition probabilities after k time units. That is, $M^k_{i,j}$ is the probability that a link at state i transits to state j after k time units. Therefore, M is a $C(n+2) \times C(n+2)$ matrix. The formal derivation of $C(n)$ can be found in [36]:

$$C(n) = \begin{cases} 1 & n = 0 \\ \frac{(n+1)(n+3)}{4} & n > 0 \text{ and n is odd} \\ \frac{n(n+4)}{4} + 1 & n > 0 \text{ and n is even} \end{cases} .$$

Suppose that a wireless link is in state i at time 0. The probability that the link will become broken at time t is

$$P_1(i,t) = \sum_{j \in \text{layer } n+1,\, n+2} M^t_{i,j}.$$

The probability that the wireless link is alive at time $t-1$ but becomes broken at time t is

$$P_2(i,t) = \begin{cases} P_1(i,t) & \text{if } t = 1 \\ P_1(i,t) - P_1(i,t-1) & \text{if } t > 1 \end{cases} .$$

Now consider an α-hop route $R = [s_1, s_2, \cdots, s_\alpha]$, where $s_i, i = 1..\alpha$, is the state of the ith wireless link in R. The probability that R is still alive after t time units is

$$P_3(R,t) = \prod_{i=1}^{\alpha} (1 - P_1(s_i,t)).$$

A path breaks when one or more of its links break. So the probability that R becomes broken after t time units is

$$P_4(R,t) = 1 - P_3(R,t)$$

and the probability that R is alive at time $t-1$ but becomes broken at time t is

$$P_5(R,t) = \begin{cases} P_4(R,t) & \text{if } t = 1 \\ P_4(R,t) - P_4(R,t-1) & \text{if } t > 1 \end{cases} .$$

Let each wireless LAN card support m rates, $R_1, R_2, \ldots, R_m$, such that rate $R_i, i = 1..m$, will be used if the destination host falls between (including) layers $n_{i-1}+1$ and n_i from the source, where $n_0 = -1$ and $n_m = n$. For example, reference [14] models an IEEE 802.11b card by $R_1 = 11$, $R_2 = 5.5$, $R_3 = 2$, $R_4 = 1$, $n_0 = -1$, $n_1 = \lfloor \frac{n}{2} \rfloor$, $n_2 = \lfloor \frac{2n}{3} \rfloor$, $n_3 = \lfloor \frac{5n}{6} \rfloor$, and $n_4 = n$. Given the initial state of link $s_i, i = 1..\alpha$, the probability that the link's rate falling in R_j (i.e., the link's distance is between layers $n_{j-1}+1$ and n_j) at time t is

$$P_6(s_i, R_j, t) = \sum_{k \in \text{ layer } (n_{j-1}+1)..n_j} M^t_{i,k}.$$

Therefore, the bandwidth of R at time t can be modeled by summing the expected transmission rate of the route over all possible rate combination of links in R at time t as follows

$$B(R,t) = \sum_{i_1=1}^{m} \sum_{i_2=1}^{m} \cdots \sum_{i_\alpha=1}^{m} P_6(s_1, R_{i_1}, t)$$
$$\times P_6(s_2, R_{i_2}, t) \times \cdots \times P_6(s_\alpha, R_{i_\alpha}, t) \times f(R_{i_1}, R_{i_2}, \cdots, R_{i_\alpha}),$$

where the function f is the transmission rate of the route. It will be estimated in next subsection. Finally, the expected throughput of route R, denoted by $E(R)$, can be derived by summing the expected route throughput over all possible route lifetime of R as follows

$$E(R) = \sum_{t_1=1}^{\infty} \left(P_5(R, t_1) \times \sum_{t_2=1}^{t_1} \frac{B(R, t_2)}{t_1} \right). \tag{3.1}$$

3.3.1 Estimation of the Function $f(\cdot)$

In this subsection, we will propose a method to estimate the throughput of a given α-hop route $R = [s_1, s_2, \cdots, s_\alpha]$, where $s_i, i = 1, 2, .., \alpha$, is the state of the ith wireless link. Recall that we represent link state as a vector in a 2-dimensional space. So from each s_i, we can derive the distance between the two endpoints of the link and the most appropriate rate r_i that should be used by this link. Given such a route R, our goal is to derive its transmission

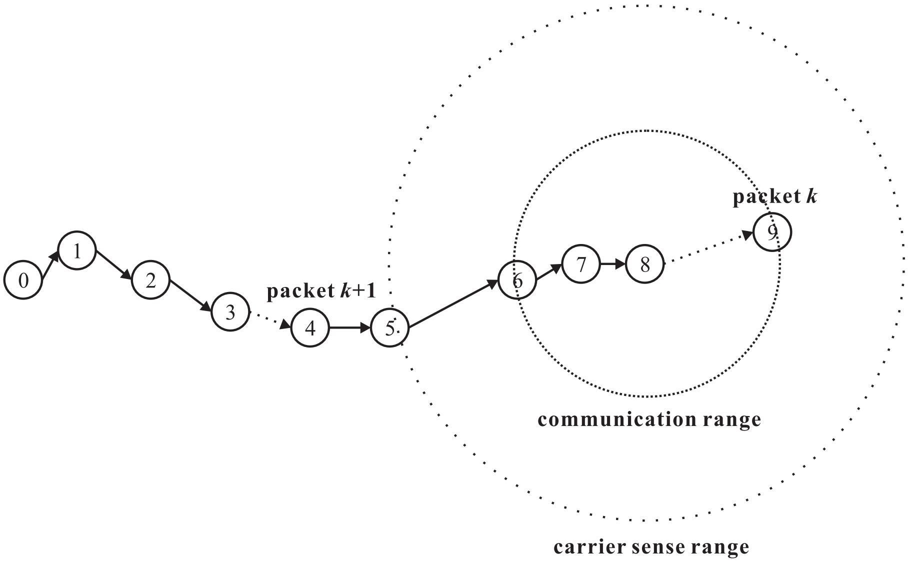


Figure 3.5: The 9-hop route with its most interfered region including host $5 \sim 9$.

rate $f(r_1, r_2, \cdots, r_\alpha)$. An ideal channel condition is assumed in the estimation such that a transmission fails only when collisions occur.

The hosts in routing path R are numbered from 0 to α such that host 0 is the traffic source and host α the sink of the path. Therefore, s_i is the state of the link between host $i-1$ and host i. Except the sink host, we can assign each host i in R an interference group G_i, which contains host i and each host j in front of i (i.e., $i > j$) that can sense the signal of i when i is transmitting. Intuitively, hosts in the same group will not transmit at the same time, but hosts in different groups may be allowed to transmit simultaneously. Note that the interference group G_i is defined to make hosts in G_i can not transmit at the same time. If hosts behind host i are included in G_i, hosts in front of host i and hosts behind host i may be allowed to transmit simultaneously. In our estimation, we model the function $f(\cdot)$ by

$$f(r_1, r_2, \cdots, r_\alpha) = \frac{1}{\max_{i=1\sim(\alpha-1)}\{\frac{1}{r_{(i+1)}} + \sum_{j \,\in\, G_i} \frac{1}{r_j}\}},$$

where $\max_{i=1\sim(\alpha-1)}\{\frac{1}{r_{(i+1)}} + \sum_{j \,\in\, G_i} \frac{1}{r_j}\}$ is the time required to transmit a bit along R in the most interfered region.

The basic concept of our modulation is that host a receiving packet $k+1$ can not be in the carrier sense range of host b sending packet k. In other words, these two packets can be transmitted simultaneously if host a is not in the carrier sense range of host b. From the view of pipelining, when packet k arrived at host b, packet $k+1$ has arrived at the host near but out of the carrier sense range of host a. Since the slowest stage of a pipeline dominates its throughput, we take the time T of travelling through the most interfered region as the transmission time for packets in R. So every packet except the first one in R only takes T to arrive at sink host α after its previous packet arrived at the sink host. Therefore, the expected transmission time for each packet of size S being transmitted in R is $T = \max_{i=1\sim(\alpha-1)}\{\frac{S}{r_{(i+1)}} + \sum_{j \in G_i} \frac{S}{r_j}\}$, and the transmission rate of R is $\frac{S}{T} = \frac{1}{\max_{i=1\sim(\alpha-1)}\{\frac{1}{r_{(i+1)}} + \sum_{j \in G_i} \frac{1}{r_j}\}}$. For example, Fig. 3.5 shows the 9-hop route with its most interfered region including host $5 \sim 9$. The links that can transmit concurrently are indicated by dashes. We can find that when packet k arrived at sink host 9, packet $k+1$ has arrived at host 4. It is because that when host 3 is sending packet $k+1$ to host 4, host 8 can send packet k to sink host 9 at the same time without interfering the receiving of host 4. Therefore, packet $k+1$ in the 9-hop route only takes the time of travelling through hosts $5 \sim 8$ to arrive at sink host 9 after packet k arrived at the sink host. Accordingly, the transmission rate of the 9-hop route is $\frac{1}{\frac{1}{r_9} + \sum_{j \in G_8} \frac{1}{r_j}} = \frac{1}{\frac{1}{r_5}+\frac{1}{r_6}+\frac{1}{r_7}+\frac{1}{r_8}+\frac{1}{r_9}}$.

3.4 Simulation Results

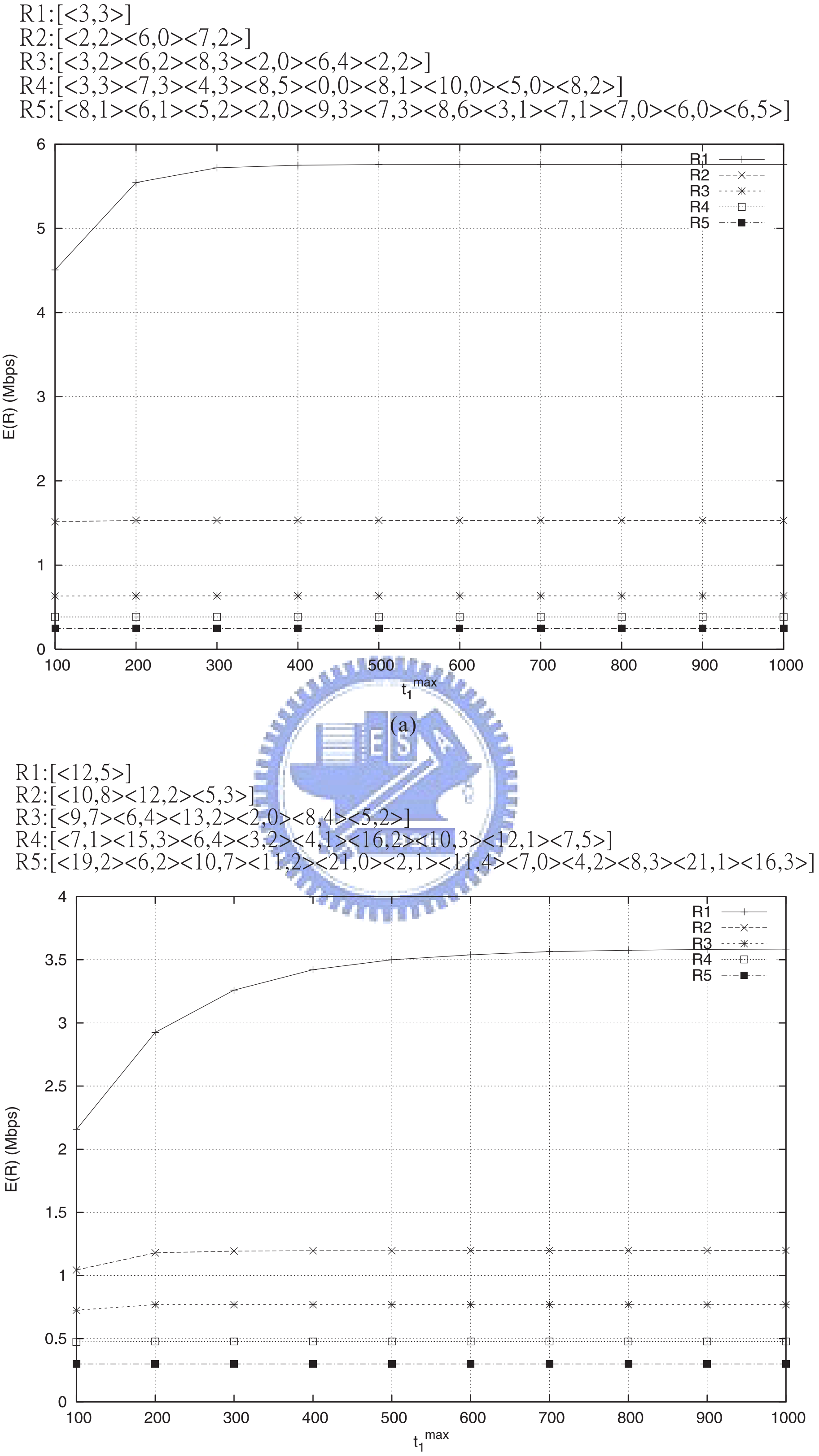
In this section, we present our simulation results. Most current products of IEEE 802.11b have a transmission distance of $150 \sim 300$ meters. We set the radius of each hexagonal cell to 10 meters, so hosts' transmission range is around $n = 15 \sim 30$ layers. The carrier sense range is set to be the same with the transmission range, and the mobile host is set to randomly select its roaming direction per time unit. Each time unit is set to 10 seconds, so as to model the roaming speed of pedestrians (around 1 m/s). The saturated traffic and unlimited buffer are used in our simulation, and the roaming speed of each mobile host is set to 1 m/s.

First, we try to determine the level of accuracy. Observe that index t_1 in Eq. (1) ranges from 1 to infinity. This is computationally infeasible. So we need to determine an upper bound for t_1 (called t_1^{max} below). We randomly generate five routing paths with 1, 3, 6, 9, and 12 hops, respectively. We calculate their expected throughputs by varying t_1^{max} from 100 to 1000. The results are in Fig. 3.6 for $n = 15$ and 25, respectively. Since $E(R)$ stablizes at $t_1^{max} \approx 300$, we will set $t_1^{max} = 1000$ in the rest of the simulations.

Our results can be used to help route selection in a MANET. Hop count is probably the most widely used route selection criterion. Our result may provide an alternative choice if throughput is the main concern, especially under a multi-rate environment. We pick a source cell and a destination cell, and place some relay hosts between them which are separated uniformly. We evaluate the expected route throughput by varying the number of relay hosts (and thus path length that is the number of links in the path). Fig. 3.7 shows our results for $n = 15$ and 25, respectively.

In both cases, we see that the throughput increases with the path length initially, but decreases afterwards after certain thresholds. In fact, there are two contradicting factors here. A very small path length implies a low transmission rate in each hop, thus leading to low path throughput. On the contrary, a longer path implies potentially higher rates and the higher degree of spectral reuse, but may risk a higher probability of existence of low-rate links in the path (thus becoming a bottleneck). Our result may be used here to make a smart choice.

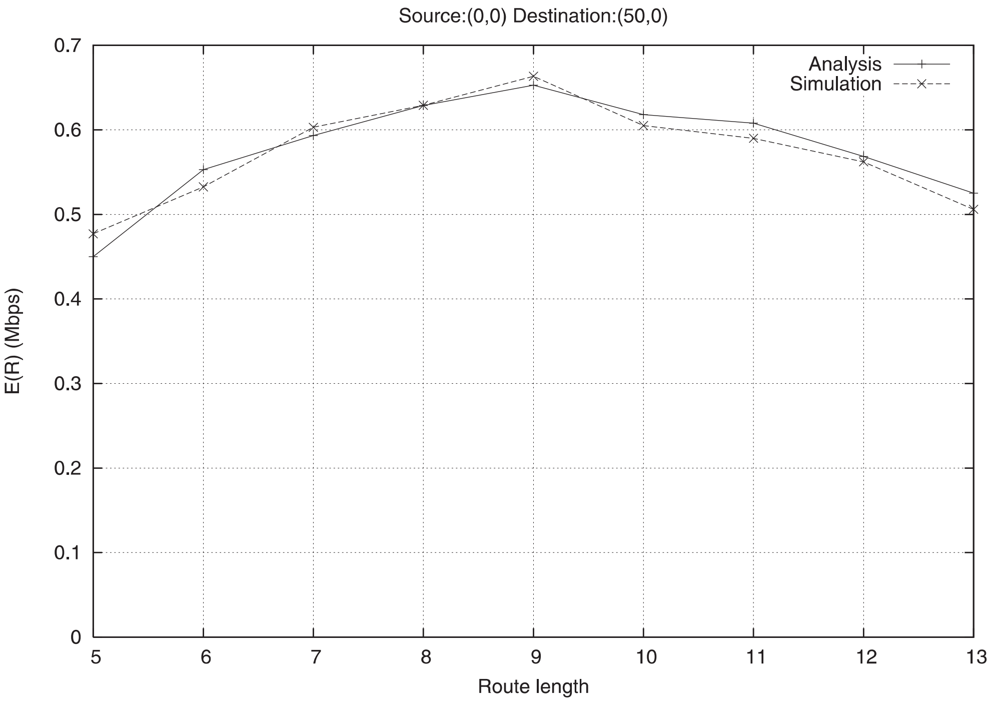
Fig. 3.7 also contains comparisons of simulation and analytical results. In each simulation, we evaluate the throughput of the path every time unit until it is broken and then calculate the average throughput. Each simulation is repeated 20,000 times to capture the random roaming of mobile hosts, and then we take the average throughput. As can be seen, the simulated and analytical results are quite close, which justifies the correctness of our derivation.



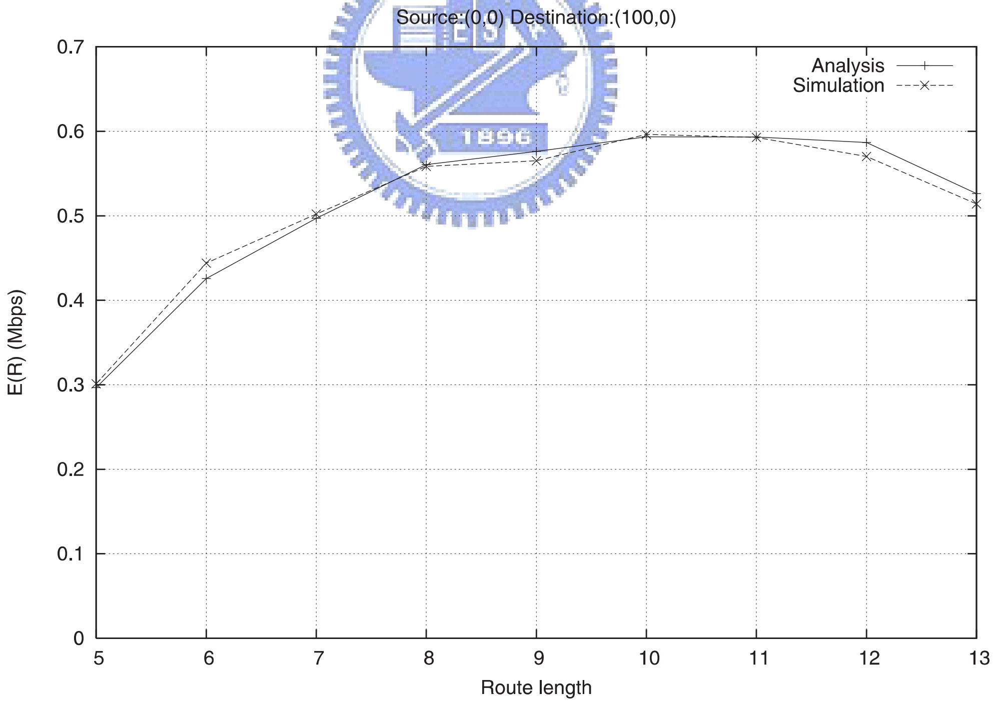
(a)

(b)

Figure 3.6: Expected route throughput vs. t_1^{max}: (a) $n = 15$ and (b) $n = 25$.



(a)



(b)

Figure 3.7: Expected route throughput vs. path length: (a) $n = 15$ and (b) $n = 25$.

Chapter 4

Design and Analysis of Contention-based Request Schemes for Best-Effort Traffics in IEEE 802.16 Networks

4.1 Motivations

In this work, we studies the collision-resolution mechanisms for transmitting uplink BE requests to the BS. The request scheme defined in the standard is compared against the proposed *Request Piggyback* (RPB) scheme. Reference [37] proposes to transmit DL-MAP and UL-MAP control messages on data packets with high data rates to reduce MAC overhead. Optimal contention periods for single and multiple classes of flow priorities are studied in [38,39]. However, these studies do not study the request scheme itself. Reference [40] models the *truncated binary exponential backoff* (TBEB) scheme in 802.16 assuming that each SS always has traffic to be sent to the BS and each request message only asks for one data slot. The proposed RPB scheme is shown to outperform the TBEB scheme.

4.2 The Request Piggyback Scheme

In this section, a new scheme for sending requests of uplink BE traffics is proposed. We first review the TBEB scheme in IEEE 802.16. Fig. 4.1 illustrates an IEEE 802.16 TDD frame.

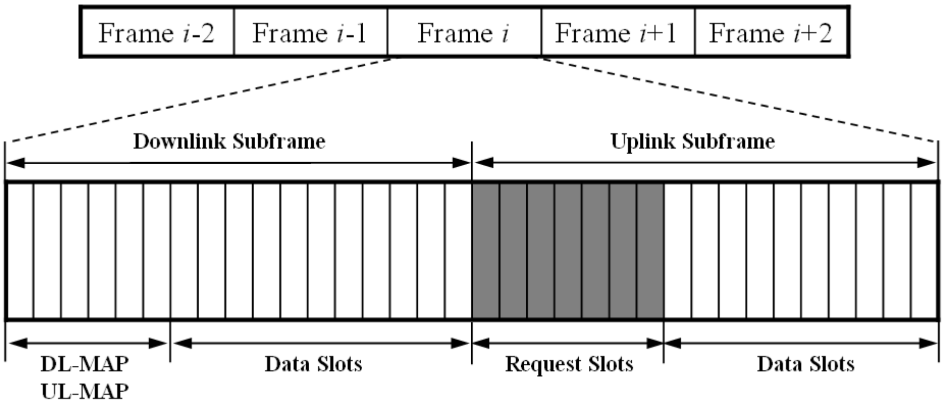


Figure 4.1: The TDD frame structure defined in IEEE 802.16.

When a SS has uplink BE traffics, it sets its initial backoff window to W_0 and randomly selects a backoff counter within this window. After the counter expires, it transmits its request. If the request succeeds, the BS will allocate bandwidths to the SS. Otherwise, the SS multiplies its backoff window by a factor of two, as long as it does not exceed the maximum value W_{max}. Then, it repeats the process until either the request succeeds or the maximum number of retries is reached.

The main problem with the TBEB scheme is that the number of waiting frames for a successful request may increase rapidly after consecutive collisions. So SSs may suffer long delay due to accidental consecutive collisions, leading to unfairness of bandwidth allocation. To alleviate this problem, we propose to allow a SS to piggyback its new BE queue length if it still has uplink burst(s) to its BS. This would even reduce the chance of contention. If the SS does not have new buffered packets, the piggyback mechanism is not taken. When an idle SS has new packets to be sent, it has to go through the same backoff and contention procedure as defined in IEEE 802.16.

Since we believe that the above piggyback mechanism can significantly reduce the possibility of collision, we suggest that the backoff window can be kept constant, equal to the number of request slots and does not need to be doubled after each collision. A collided request message can be immediately retransmitted in the next frame. For piggybacking requests, the Grant Management subheader (16 bits) in a generic MAC frame can be used. The number of bytes of the bandwidth request is incremental and is limited to 2^{16} bytes.

4.3 Analytical Results

This section analyzes the request success probability and the packet delivery delay of the TBEB and the RPB schemes. In our analysis, we assume that a SS sends at most one request in a frame and multiple BE connections in a SS are treated as a single, aggregated connection for simplicity. Suppose that there are n SSs under a BS and s request slots per frame. For TBEB, we set $W_0 = s$ and $W_{max} = 2^m W_0$, where m is the maximum number of retries. This means that in the ith retrial, a SS will send its request randomly from one of the upcoming 2^i frames.

To derive the expected number of contending SSs per frame, we first calculate the probability p_{tx} that a SS will transmit a request message in a frame when it has packets to send. It can be obtained by computing the average number n_{tx} of requests transmitted in a TBEB process and the average period n_{tf} (in unit of frames) of a TBEB process. We have

$$\begin{aligned} n_{tx} &= \sum_{i=1}^{m+1} i \times Prob(\text{request sent exactly } i \text{ times}) \\ &= \sum_{i=1}^{m} i(1-c)c^{i-1} + (m+1)c^m, \end{aligned}$$

where c is the request collision probability. The average period of a TBEB process depends on the number of request retries and the backoff counter which is randomly selected in the beginning of a backoff process. For the ith retry, the backoff window size is $W_i = 2^{i-1}W_0$. The average number of frames n_{af} (which is a function of W_i) that the ith retry needs to be deferred is calculated as

$$n_{af}(W_i) = \frac{1}{W_i}\sum_{j=1}^{W_i} j.$$

Thus, the average period of a TBEB process can be modelled by summing the expected number of deferring frames over all possible numbers of retries:

$$n_{tf} = \sum_{i=1}^{m}(1-c)c^{i-1}\left(\sum_{j=1}^{i} n_{af}(W_j)\right) + c^m \sum_{i=1}^{m+1} n_{af}(W_i).$$

This gives the expected number of contending SSs per frame

$$n_{req}^{\text{TBEB}} = n \cdot p_{tx} \cdot (1 - e^{-\lambda n_{tf} f}),$$

where

$$p_{tx} = \frac{n_{tx}}{n_{tf}},$$

f is the frame duration, and λ is the packet arrival rate with poisson inter-arrival times. The probability that a SS successfully transmits its request in a frame is

$$p_s^{\text{TBEB}} = (\frac{s-1}{s})^{n_{req}^{\text{TBEB}}-1},$$

and the expected number of frames that a SS has to wait before a successful request is submitted is

$$d_s^{\text{TBEB}} = n_{tf}.$$

Let M be the number of frames allocated to transmit the data of a request, which is a constant controlled by the BS. The packet delivery delay per data request (in unit of frame) can be written as

$$d_{pkt}^{\text{TBEB}} = 1 + d_s^{\text{TBEB}} + M = 1 + n_{tf} + M.$$

For the RPB scheme, a Markov model is derived. We first define the possible states of a SS:

- $IDLE$: The SS has no BE traffic currently.

- REQ: The SS is contending for a request slot.

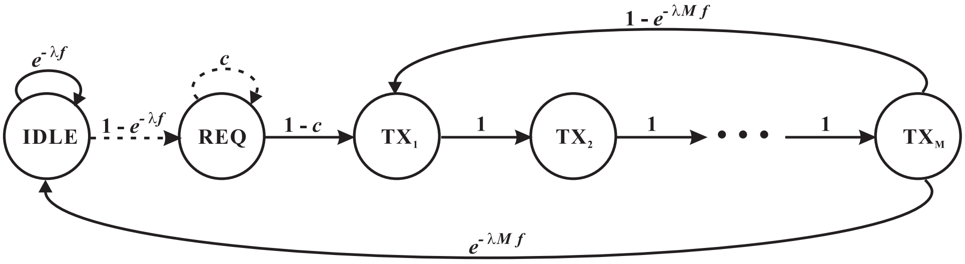


Figure 4.2: The state transition diagram of a SS under the RPB model.

- $TX_i, i = 1..M$: Bandwidth has been allocated for the SS and it has been served for i frames.

The state-transition diagram is shown in Fig. 4.2. The probability associated with each transition can be obtained from the frame duration f, the request collision probability c, and the packet arrival rate λ with poisson inter-arrival times. There are two events which will trigger a SS to contend for request slots: (i) the SS switches from the idle state to the request state as a new packet arrives, and (ii) the SS stays in the request state as a collision was experienced previously. These two events are illustrated in Fig. 4.2 by dashes.

Next, we compute the probability that the SS will stay in each state. Let $Prob(x)$ be the probability that it is in state x. Since the sum of probabilities over all states must be 1, we have

$$Prob(IDLE) + Prob(REQ) + \sum_{i=1}^{M} Prob(TX_i) = 1.$$

Considering the equilibrium of flows for state $IDLE$, we have

$$Prob(IDLE)(1 - e^{-\lambda f}) = Prob(TX_M) \cdot e^{-\lambda M f}.$$

Similarly, from the equilibrium of flows for state REQ,

$$Prob(REQ)(1 - c) = Prob(IDLE)(1 - e^{-\lambda f}),$$

and for state TX_1,

$$Prob(TX_1) = Prob(REQ)(1 - c) + Prob(TX_M)(1 - e^{-\lambda M f}).$$

For state TX_i, $i = 2..M$, we have

$$Prob(TX_i) = Prob(TX_{i-1}).$$

There are $M + 2$ state probabilities to be determined. From the above equations, we can obtain that

$$\begin{aligned}
Prob(IDLE) &= \frac{e^{-\lambda Mf}(1-c)}{D} \\
Prob(REQ) &= \frac{(1-e^{-\lambda f})e^{-\lambda Mf}}{D} \\
Prob(TX_1) &= \frac{(1-e^{-\lambda f})(1-c)}{D} \\
Prob(TX_M) &= Prob(TX_{M-1}) = \cdots = Prob(TX_1),
\end{aligned}$$

where $D = e^{-\lambda Mf}(1-c) + (1-e^{-\lambda f})e^{-\lambda Mf} + M(1-e^{-\lambda f})(1-c)$. Thus, the expected number of SSs to contend for request slots per frame can be derived as

$$\begin{aligned}
n_{req}^{\text{RPB}} &= n \cdot [(1-e^{-\lambda f})Prob(IDLE) + c \cdot Prob(REQ)] \\
&= n \cdot [(1-c)Prob(REQ) + c \cdot Prob(REQ)] \\
&= n \cdot Prob(REQ).
\end{aligned}$$

Next, we calculate the expected number d_s^{RPB} of frames that a SS has to wait before a successful request. Let the probability to send a request without collision be p_s^{RPB}. We can derive that

$$p_s^{\text{RPB}} = (\frac{s-1}{s})^{n_{req}^{\text{RPB}}-1}$$

and

$$d_s^{\text{RPB}} = \frac{1}{p_s^{\text{RPB}}} = (\frac{s}{s-1})^{n_{req}^{\text{RPB}}-1}.$$

The delivery delay d_{pkt}^{RPB} of a packet is the period from its arrival at the queue to its complete delivery to the BS. So d_{pkt}^{RPB} consists of three components: (i) the request delay d_{req} (the expected number of waiting frames for a successful request), (ii) the piggyback delay d_{pg} (the expected number of waiting frames for a new bandwidth allocation if the SS is currently in a transmission state TX_i, $i = 1..M$), and (iii) the packet transmission time t_{tx} (M frames controlled by the BS). Component (ii) can be derived as

$$d_{pg} = \sum_{i=1}^{M} i \cdot Prob(TX_{M-i+1}) = \frac{M(M+1)}{2} Prob(TX_1).$$

Summing all three components, the packet delivery delay is

$$\begin{aligned} d_{pkt}^{\text{RPB}} &= d_{req} + d_{pg} + t_{tx} \\ &= \begin{cases} 1 + d_s^{\text{RPB}} + M & \text{the SS is in } IDLE \text{ state} \\ \frac{d_s^{\text{RPB}}}{2} + M & \text{the SS is in } REQ \text{ state} \\ (M + 1 - i) + M & \text{the SS is in } TX_i \text{ state} \end{cases} \\ &= (1 + d_s^{\text{RPB}}) Prob(IDLE) + \frac{d_s^{\text{RPB}}}{2} Prob(REQ) \\ &\quad + \frac{M(M+1)}{2} Prob(TX_1) + M. \end{aligned}$$

4.4 Simulation Evaluation

We verify the derived request success probability and packet delivery delay by a C++ simulator. The frame duration is set to 5 ms, and the request collision probability is obtained through our simulation. In all figures, lines are mathematical results, and symbols represent simulation results. Clearly, the mathematical results fit quite well with the simulation results.

As shown in Fig. 4.3, when the number of SSs increases, the request success probability decreases. For TBEB, the request success probability decreases rapidly due to more contentions. It can be observed that RPB has a higher request success probability than TBEB. Even with $M = 3$, RPB's request success probability is still about 0.5 when there are 100 SSs associated with the BS.

Fig. 4.4 shows the packet delivery delay for various numbers of SSs. While the delay of TBEB increases exponentially, the delay of RPB only increases linearly as the number of SSs increases. The same observation holds for all other cases of $\lambda = 10, 20, ..., 100$ and

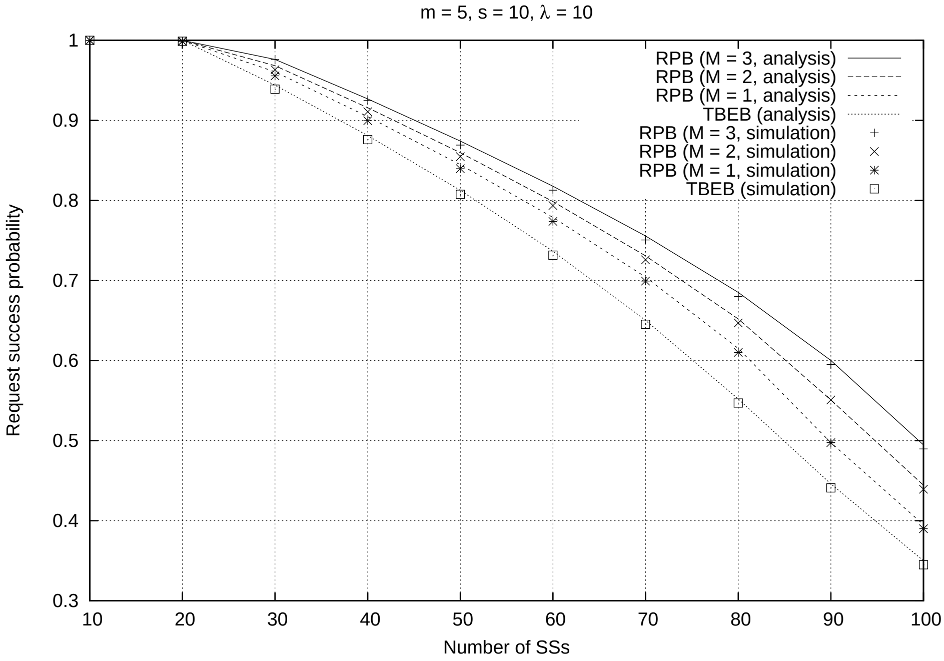


Figure 4.3: Comparison of request success probabilities.

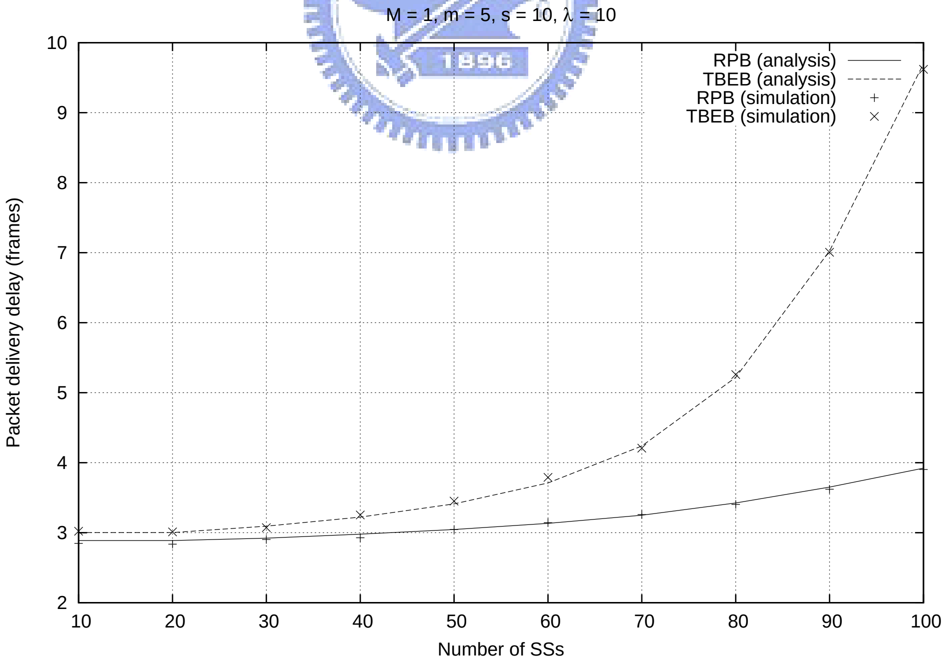


Figure 4.4: Comparison of packet delivery delays.

$M = 1, 2, ..., 10$. In Fig. 4.4, TBEB has a much higher delay than RPB since its number of deferring frames are much higher. RPB produces a lower delay for packet delivery because SSs transmitting data can piggyback their new bandwidth requests without waiting any frame.

From these results, we conclude that the RPB scheme can achieve a higher request success probability and a lower packet delivery delay, leading to more efficient use of wireless bandwidths. On the other word, adopting our scheme in IEEE 802.16 networks can both avoid the BS wasting bandwidth due to insufficient received requests and prevent SSs from buffer overflowing caused by numerous delayed packets.

Chapter 5

Conclusions and Future Directions

In Chapter 2, we have shown how to increase the degree of spectral reuse in an IEEE 802.16 mesh network. An integrated spectral reuse framework for centralized scheduling and a routing tree construction scheme are developed. Compared to previous works, our framework is most complete in exploiting spectral reuse of IEEE 802.16 mesh networks in the sense that it takes dynamic traffic loads of SSs into account and integrates not only a bandwidth scheduling scheme but also a time-slot allocation scheme. In addition, a routing algorithm with tree optimization is proposed. We have also developed an extension of our framework to support bandwidth requirements of real-time flows. Simulation results have shown that the proposed framework significantly improves the network throughput and the flow granted ratio compared with the specification in the IEEE 802.16 standard.

In Chapter 3, we have shown how to formulate the throughput given a path in which hosts roam around in a random walk model and the communication interfaces have the rate adaptive capability. As far as we know, this issue has not been carefully studied yet. Simulation results show that the proposed formulation can be used to evaluate path throughput accurately. We believe that the path throughput is a better metric than the traditional metrics, such as the hop count, for route selection in multi-rate ad hoc networks and that the proposed mechanism can be easily embedded into most of the current routing protocols for mobile ad hoc networks.

In Chapter 4, we have studied two contention-based request schemes for BE traffics in

IEEE 802.16 networks, which are heavily affecting the performance of BWA systems. Analytical and simulation results show that our proposed scheme can significantly release request slot contention and efficiently minimize packet delivery delay. In particular, our investigation is useful for evolving a better request/grant mechanism for IEEE 802.16 networks.

Based on the results presented above, several directions may deserve further investigation. First, more QoS factors of real-time flows such as delay constraints and jitters could be considered in the slot assignment strategy [41]. Second, flow differentiation rater than flow prioritization could be considered in the bandwidth allocation scheme to prevent non-real-time flows from starvation. Third, multi-path routing and distributed scheduling could be considered to provide better performance. Finally, the limitation that a slot is only exclusively used for uplink or downlink throughout the whole network could be relaxed for better bandwidth efficiency.

Bibliography

[1] H. Y. Wei, S. Ganguly, R. Izmailov, and Z. Haas. Interference-aware IEEE 802.16 WiMAX mesh networks. In *IEEE Vehicular Technology Conference*, volume 5, pages 3102–3106, 2005.

[2] L. Fu, Z. Cao, and P. Fan. Spatial reuse in IEEE 802.16 based wireless mesh networks. In *IEEE International Symposium on Communications and Information Technology*, volume 2, pages 1358–1361, 2005.

[3] J. Tao, F. Liu, Z. Zeng, and Z. Lin. Throughput enhancement in WiMAX mesh networks using concurrent transmission. In *IEEE International Conference on Wireless Communications, Networking and Mobile Computing*, volume 2, pages 871–874, 2005.

[4] *IEEE Standard 802.16-2004, IEEE Standard for Local and Metropolitan Area Networks Part 16: Air Interface for Fixed Broadband Wireless Access Systems*, 2004. IEEE.

[5] A. Ghosh, D. R. Wolter, J. G. Andrews, and R. Chen. Broadband wireless access with WiMAX/802.16: Current performance benchmarks and future potential. *IEEE Communications Magazine*, 43(2):129–136, 2005.

[6] C. Eklund, R. B. Marks, K. L. Stanwood, and S. Wang. IEEE Standard 802.16: A technical overview of the WirelessMAN$^{\text{TM}}$ air interface for broadband wireless access. *IEEE Communications Magazine*, 40(6):98–107, 2002.

[7] I. F. Akyildiz, X. Wang, and W. Wang. Wireless mesh networks: a survey. *Computer Networks and ISDN Systems*, 47(4):445–487, 2005.

[8] D. Johnson, D. Maltz, and J. Broch. DSR: The dynamic source routing protocol for multihop wireless ad hoc networks. *Ad Hoc Networking, Chapter 5, Addison-Wesley*, 2000.

[9] R. Dube, C. Rais, K. Wang, and S. Tripathi. Signal stability based adaptive routing (SSA) for ad-hoc mobile networks. *IEEE Personal Communications*, 4(1):36–45, Feb. 1997.

[10] C. K. Toh. Associativity-based routing for ad hoc mobile networks. *Wireless Personal Communications Journal*, 4(2):103–139, March 1997.

[11] S. Singh, M. Woo, and C. S. Raghavendra. Power-aware routing in mobile ad hoc networks. In *ACM/IEEE MOBICOM*, pages 181–190, 1998.

[12] R. Draves, J. Padhye, and B. Zill. Comparison of routing metrics for static multi-hop wireless networks. In *SIGCOMM'04*, Aug. 2004.

[13] D. De Couto, D. Aguayo, J. Bicket, and R. Morris. High-throughput path metric for multi-hop wireless routing. In *ACM/IEEE MOBICOM*, Sep. 2003.

[14] B. Awerbuch, D. Holmer, and H. Rubens. High throughput route selection in multi-rate ad hoc wireless networks. In *Wireless On-Demand Network Systems*, pages 251–268, Jan. 2004.

[15] A. Kamerman and L. Monteban. WaveLAN II: a high-performance wireless lan for the unlicensed band. *Bell Labs Technical Journal*, pages 118–133, Summer 1997.

[16] G. Holland, N. Vaidya, and P. Bahl. A rate-adaptive mac protocol for multi-hop wireless networks. In *ACM/IEEE MOBICOM*, pages 236–251, 2001.

[17] S. Zou, S. Cheng, and Y. Lin. Multi-rate aware topology control in multi-hop ad hoc networks. In *Wireless Communications and Networking Conference*, pages 2207–2212, Mar. 2005.

[18] V. P. Mhatre, H. Lundgren, and C. Diot. Mac-aware routing in wireless mesh networks. In *Wireless on Demand Network Systems and Services*, pages 46–49, Jan. 2007.

[19] Y. Seok, J. Park, and Y. Choi. Multi-rate aware routing protocol for mobile ad hoc networks. In *IEEE VTC*, 2003.

[20] G. Chu, D. Wang, and S. Mei. A QoS architecture for the MAC protocol of IEEE 802.16 BWA system. In *IEEE International Conference on Communications, Circuits and Systems and West Sino Expositions*, volume 1, pages 435–439, 2002.

[21] M. Hawa and D. Petr. Quality of service scheduling in cable and broadband wireless access systems. In *IEEE International Workshop on Quality of Service*, pages 247–255, 2002.

[22] K. Wongthavarawat and A. Ganz. IEEE 802.16 based last mile broadband wireless military networks with quality of service support. In *IEEE Military Communications Conference*, volume 2, pages 779–784, 2003.

[23] V. Gunasekaran and F. C. Harmantzis. Affordable infrastructure for deploying WiMAX systems: Mesh v. non mesh. In *IEEE Vehicular Technology Conference*, volume 5, pages 2979–2983, 2005.

[24] M. Cao, W. Ma, Q. Zhang, X. Wang, and W. Zhu. Modelling and performance analysis of the distributed scheduler in IEEE 802.16 mesh mode. In *ACM International Symposium on Mobile Ad Hoc Networking and Computing*, pages 78–89, 2005.

[25] S. M. Cheng, P. Lin, D. W. Huang, and S. R. Yang. A study on distributed/centralized scheduling for wireless mesh network. In *ACM International Conference on Wireless Communications and Mobile Computing*, pages 599–604, 2006.

[26] H. Shetiya and V. Sharma. Algorithms for routing and centralized scheduling to provide QoS in IEEE 802.16 mesh networks. In *ACM Workshop on Wireless Multimedia Networking and Performance Modeling*, pages 140–149, 2005.

[27] J. Chen, W. Jiao, and Q. Guo. An integrated QoS control architecture for IEEE 802.16 broadband wireless access systems. In *IEEE Global Telecommunications Conference*, volume 6, pages 3330–3335, 2005.

[28] B. Johansson P. Soldati and M. Johansson. Distributed optimization of end-to-end rates and radio resources in WiMAX single-carrier networks. In *IEEE Global Telecommunications Conference*, 2006.

[29] D. Kim and A. Ganz. Fair and efficient multihop scheduling algorithm for IEEE 802.16 BWA systems. In *IEEE International Conference on Broadband Networks*, pages 833–839, 2005.

[30] M. Udi. *Introduction to Algorithms: A Creative Approach*. Addison-Wesley Publishing Company, 1989.

[31] NS-2. The network simulator. http://www.isi.edu/nsnam/ns/.

[32] I. Akyildiz and J. Ho and Y. Lin. Movement-based location update and selective paging for PCS networks. *IEEE/ACM Trans. on Networking*, 4(4):629–638, Aug. 1996.

[33] I. F. Akyildiz and J. S. M. Ho. Dynamic mobile user location update for wireless PCS networks. *ACM Wireless Networks*, 1(2):187–196, July 1995.

[34] J. S. M. Ho and I. F. Akyildiz. Mobile user location update and paging under delay constraints. *ACM Wireless Networks*, 1(4):413–426, Dec. 1995.

[35] Y. C. Tseng, Y. F. Li, and Y. C. Chang. On the lifetime of routing paths in multi-hop mobile ad hoc networks. *IEEE Trans. on Mobile Computing*, 2(4):366–376, Oct.-Dec 2003.

[36] Y. C. Tseng, and W. N. Hung. An improved cell type classification for random walk modeling in cellular networks. *IEEE Communication Letters*, 5(8):337–339, Aug. 2001.

[37] J.Y. Kim and D.-H. Cho. Piggybacking Scheme of MAP IE for Minimizing MAC Overhead in the IEEE 802.16e OFDMA Systems. In *IEEE 66th Vehicular Technology Conference 2007 (VTC-2007 Fall)*, pages 284–288, Sept. 2007.

[38] J. Yan and G.-S. Kuo. Cross-layer Design of Optimal Contention Period for IEEE 802.16 BWA Systems. In *IEEE International Conference on Communications 2006*, volume 4, pages 1807–1812, June 2006.

[39] S.-M. Oh and J.-H. Kim. The analysis of the optimal contention period for broadband wireless access network. In *Third IEEE International Conference on Pervasive Computing and Communications Workshops 2005 (PerCom 2005 Workshops)*, pages 215–219, Mar. 2005.

[40] J. He, K. Guild, K. Yang, and H.-H. Chen. Modeling Contention Based Bandwidth Request Scheme for IEEE 802.16 Networks. *IEEE Communications Letters*, 11(8):698–700, Aug. 2007.

[41] P. Djukic and S. Valaee. Link scheduling for minimum delay in spatial re-use TDMA. In *IEEE INFOCOM*, pages 28–36, 2007.

Vita

Lien-Wu Chen

Department of Computer Science
National Chiao Tung University
1001 Ta Hsueh Road, Hsinchu, Taiwan 300
Email: lwchen@cs.nctu.edu.tw

Education

Ph.D.: Computer Science, National Chiao Tung University (2003.9 ∼ 2008.12)

M.S.: Computer Science and Information Engineering, National Central University (1998.9 ∼ 2000.6)

B.S.: Computer Science and Information Engineering, Fu Jen Catholic University (1994.9 ∼ 1998.6)

Publication Lists

Journal papers

1. Yu-Chee Tseng, Lien-Wu Chen, Ming-Hour Yang, and Jan-Jan Wu, “A Stop-or-Move Mobility Model for PCS Networks and Its Location-Tracking Strategies”, *Computer Communications*, Vol. 26, No. 12, July 2003, pp. 1288-1301. (SCIE, EI)

2. Lien-Wu Chen and Yu-Chee Tseng, “Design and Analysis of Contention-based Request Schemes for Best-Effort Traffics in IEEE 802.16 Networks”, *IEEE Communications Letters*, Vol. 12, No. 8, Aug. 2008, pp. 602-604. (SCI, EI)

3. Lien-Wu Chen, Yu-Chee Tseng, You-Chiun. Wang, Da-Wei Wang, and Jan-Jan Wu, “Exploiting Spectral Reuse in Routing, Resource Allocation, and Scheduling for IEEE 802.16 Mesh Networks”, *IEEE Transaction on Vehicular Technology*, to appear. (SCI, EI)

4. Lien-Wu Chen, Weikuo Chu, Yu-Chee Tseng, and Jan-Jan Wu, “Route Throughput Analysis with Spectral Reuse for Multi-Rate Mobile Ad Hoc Networks”, *Journal of Information Science and Engineering*, to appear. (SCIE, EI)

Conference papers

1. Ming-Hour Yang, Lien-Wu Chen, Yu-Chee Tseng, and Jang-Ping Sheu, “A Traveling Salesman Mobility Model and Its Location Tracking in PCS Networks”, *International Conference on Distributed Computing Systems*, 2001, pp. 517-523.

2. Yu-Chee Tseng, Weikuo Chu, Lien-Wu Chen, and Chih-Min Yu, “Route Throughput Analysis for Mobile Multi-Rate Wireless Ad Hoc Networks”, *Broadband Wireless Networking Symposium (BroadNet)*, 2004.

3. Lien-Wu Chen, Yu-Chee Tseng, Da-Wei Wang, and Jan-Jan Wu, “Exploiting Spectral Reuse in Resource Allocation, Scheduling, and Routing for IEEE 802.16 Mesh Networks”, *IEEE Vehicular Technology Conference (VTC)*, 2007-Fall.