

國立交通大學

資訊科學與工程研究所

博士論文

行動寬頻網路之省電與資源管理

Power and Resource Management
in Mobile Broadband Networks



研究生：陳建志

指導教授：曾煜棋 博士

中華民國九十八年十月

行動寬頻網路之省電與資源管理

Power and Resource Management in Mobile Broadband Networks

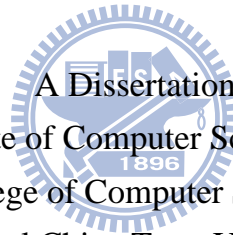
研究生：陳建志

Student : Jen-Jee Chen

指導教授：曾煜棋 博士

Advisor : Dr. 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 Institute of Computer Science and Engineering
College of Computer Science
National Chiao Tung University
in partial Fulfillment of the Requirements
for the Degree of
Doctor of Philosophy
in
Computer Science

October 2009

Hsinchu, Taiwan, Republic of China

中華民國九十八年十月

行動寬頻網路之省電與資源管理

學生：陳建志

指導教授：曾煜棋 博士

國立交通大學
資訊科學與工程研究所博士班

摘 要

近年來，無線寬頻存取越來越受歡迎，在所有的無線數據存取技術裡，IEEE 802.11 無線區域網路提供一個簡單而且省錢的方式讓使用者可以自己建立他們的區域網路，而 IEEE 802.16e 無線都會網路則提供一個無線存取方案來替代目前以有線電纜作為最後一哩存取網際網路的方式。對於行動無線用戶而言，高速寬頻、服務品質保證、連續而無縫的網路存取、與無線終端設備的續航力一直都是在無線網路上的重要議題，本篇論文專注於行動寬頻網路之省電與資源管理，主要包括三項議題：第一項議題中，我們提出一個行動寬頻網路的架構，這個架構上的行動閘道器可以同時備配多個無線介面同時連上網際網路，藉著這樣的設計，可以提供一群行動用戶無線寬頻的存取服務；第二項議題中，我們討論在行動寬頻網路上的資源管理議題，思考如何分配無線資源以及如何管理使用者的移動使得正在進行中的資料流可以獲得品質服務保證並且連線不會中斷；最後，在第三項議題中，我們討論行動寬頻網路上省電管理的問題，針對行動閘道器和行動用戶裝備 IEEE 802.16e 無線介面的情況下，我們研究其省電類別管理問題。

在第一項議題中，我們發展了一個行動寬頻網路的架構以提供一群在行動網路中的行動用戶寬頻的無線存取以及網路的行動管理，這個行動寬頻網路的網路行動管理是藉由 SIP (session initiation protocol) 來達成的，而行動網路內的使用者則以行動隨意網路的方式在內部組成一個連結的網路，這個行動隨意網路透過一個網路內的行動閘道器存取網際網路，而行動閘道器上對內裝備了數個 IEEE 802.11 介面來與內部網路連結，對外則同時裝備了多個無線介面來存取網際網路，這些對外介面

可以是 IEEE 802.11、WiMAX、PHS、或者三代或三·五代的無線介面，藉著同時整合多種不同的無線介面來連結外部網路，內部行動用戶得以享有寬頻無線存取服務。

在第二項議題中，我們首先利用 SIP 信令所夾帶的通話描述資訊設計通話允許控制以及資源管理機制，接著，考慮因為移動和無線頻道的多變性所產生的換手以及無線網路的多重速率調變，我們進一步對資源管理機制提出改善，觀察到現在的無線媒體控制層均支援服務品質機制和通話允許控制，對我們而言，整合應用層的服務品質資訊和媒體控制層的服務品質機制來設計一個跨層的資源管理機制是十分有吸引力的，所以我們設計了一考量換手和多重速率調變的跨層資源管理機制，在此機制中，當無線資源很擁擠的時候，我們可以藉由改變已存在的通話的編解碼和訊框速率動態的調整它們的資源的分配，這麼一來，不只是遭遇無線頻道品質劣化的通話其通話品質得以舒解，網路也可以接受更多的連線，除此之外，為了確保行動用戶的服務連續性，我們並且提出一個適用於包含認證加密機制的無線網路的無縫式換手機制。

在第三項議題中，我們針對的是 IEEE 802.16e 無線介面的省電類別管理問題，這個問題當行動用戶裝備的是 IEEE 802.16e 無線介面而且透過作為中繼傳播的行動開道器連到網際網路時會發生，針對 IEEE 802.16e，我們設計了四個省電類別管理演算法，四個演算法中，三個針對單一行動裝置的省電類別管理機制，一個則針對同時多個行動裝置的省電類別管理機制；對於單一行動裝置，IEEE 802.16e 定義省電類別來管理它的睡眠，但是標準並沒有描述如何為資料流定義和管理省電類別，而現存的研究僅考慮所有的資料流中最小的延遲限制來規劃單一或多個行動裝置的睡眠。針對單一行動裝置，我們提出同時使用多個省電類別來規劃行動裝置的睡眠，每一個省電類別均考慮到資料流的特性，這麼一來，我們可以為行動裝置預備更準確的資源減少浪費並且使得行動裝置可以睡得更多；同樣的動機，對於多個行動裝置的情況，我們亦設計針對每個行動裝置的服務品質特性來安排他們的睡眠，如此一來可以節省更多的能源以及更有效率的運用頻寬。

關鍵字：通話允許控制、換手、IEEE 802.11、IEEE 802.16e、行動通訊、行動

計算、行動管理、省電管理、省電類別、推播機制、服務品質、資源管理、無線寬頻存取、無線網路。



Power and Resource Management in Mobile Broadband Wireless Networks

Student: Jen-Jee Chen

Advisor: Dr. Yu-Chee Tseng

Institute of Computer Science and Engineering
National Chiao Tung University

ABSTRACT

In the recent years, wireless broadband access is gaining more popularity. Among all wireless data access technologies, the IEEE 802.11 WLANs (wireless local area networks) provide an easy and low cost solution for users to build their own local area networks while the IEEE 802.16 WMANs (wireless metropolitan area networks) provide a wireless solution to substitute the wire line last-mile Internet access. For mobile wireless users, high data rate, QoS (quality of service) guaranteed and continuous access, and long device operation time are always important issues for wireless networks. In this dissertation, we study the power and resource management in mobile broadband networks, which is composed of three major works. In the first work, we propose a mobile broadband network architecture which provides a group of mobile users broadband wireless access by attaching multiple wireless interfaces on the mobile gateway. In the second work, we discuss the resource management issue over the mobile broadband networks which considers how to manage wireless resource distribution and user mobility such that an on-going call can have guaranteed QoS and continuous connectivity. Finally, in the third work, we discuss the power management issue over the mobile broadband networks which studies the power saving class (PSC) management problem for the mobile gateway and mobile users which are equipped with IEEE 802.16e interfaces.

In the first work, we develop a mobile broadband network architecture to provide broadband wireless access and support network mobility for a group of mobile users inside the network, where the mobility management is maintained by SIP (session initiation protocol). We propose to form a mobile ad hoc network (MANET) by a group of mobile stations (MSs). The MANET is connected to the outside world via a mobile gateway, which connects to the intra MANET by some IEEE 802.11 interfaces and attaches to the Internet through more external wireless interfaces (such as IEEE 802.11, WiMAX, PHS, and 3/3.5G interfaces). By aggregating multiple external interfaces of different wireless technologies in the gateway, mobile users are allowed to have a broadband wireless access.

In the second work, we propose to design resource management mechanism by exploiting the session information carried by the SIP messages. Then, considering handoff and physical rate adaptation issues caused by mobility and wireless channel variation, we further enhance the resource management mechanism. Observing that current wireless MAC protocols all support QoS and CAC (call admission control), it is attractive to us to design a cross-layer resource management mechanism by integrating the QoS information from the application layer and the QoS mechanisms supported by the MAC layer. The proposed cross-layer scheme takes handoff and multi-rate environment into consideration. When wireless resource is stringent, we can dynamically adjust the resource distribution among existing calls by controlling their supporting codecs and frame rates. This not only takes care of calls in bad channel conditions, but also can accept more calls. In addition, to maintain continuous network continuity during handoff, we also develop a seamless post-handoff mechanism for secured wireless networks.

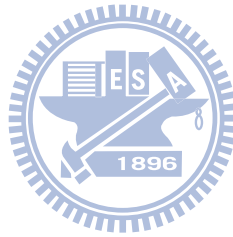
In the last work, we focus on the PSC management problem for IEEE 802.16e interfaces. This issue happens to the mobile broadband networks when the mobile gateway acts as an IEEE 802.16 relay and the mobile users are equipped with IEEE 802.16e interfaces to access the Internet via the gateway. In this part, we propose four PSC management algorithms for IEEE 802.16e wireless networks. In the four schemes, three consider a base station (BS)-MS pair, one refers to multiple MSs under a BS. For each individual MS, IEEE 802.16e defines PSCs to manage its sleep. However, the standard does not describe how to define and manage PSCs for flows. Existing works all consider only the strictest delay bound among flows to control the sleep of single or multiple MSs. Therefore, for single MS, we propose to use multiple PSCs to schedule the sleep of the MS such that the sleep schedule can more accurately capture the QoS of flows and make the MS sleep more. Based on the same motivation, for multiple MSs, we also propose to schedule each MS's sleep according to each of their QoS characteristics. These lead to less energy consumption, more efficient use of bandwidth, and more compact listening windows.

Keywords: call admission control (CAC), handoff, IEEE 802.11, IEEE 802.16, mobile communication, mobile computing, mobility management, power management, power saving class (PSC), push mechanism, quality of service (QoS), resource management, wireless broadband access, wireless network.

Acknowledgement

I would like to express my deep and sincere gratitude to my advisor, Prof. Yu-Chee Tseng for his continuous support, encouragement, and guidance throughout my Ph.D study. His extensive knowledge and creative thinking have been an invaluable help for me. Also, I would like to thank my dissertation committee members, Prof. Rong-Hong Jan, Prof. Shiao-Li Tsao, Prof. Yi-Bing Lin, Prof. Wanjiun Liao, Prof. Shiann-Tsong Sheu, and Prof. Shih-Lin Wu for their valuable comments and suggestions so that I can improve my work in the future.

Let me also say thank to the colleagues in High-Speed Communication & Computing (HSCC) laboratory for their friendship and helpful discussions. Finally, I am grateful to my dear parents, brother, sister, and girl friend for their unfailing love and firmly support in these years.



Contents

Abstract in Chinese	i
Abstract	iv
Acknowledgement	vi
Contents	vii
List of Figures	ix
List of Tables	xii
1 Introduction	1
2 Preliminaries	5
2.1 Mobility Management Protocols	5
2.2 QoS Mechanisms in Wireless Networks	15
2.3 IEEE 802.16e Sleep Mode	16
3 Resource Management Mechanisms for Mobile Broadband Networks	19
3.1 Design of a Mobile Broadband Network and Its Resource Management Mechanism	20
3.1.1 System Architecture and Motivation	20
3.1.2 Basic Operations of the Mobile Broadband Network	22
3.1.3 Proposed Push Mechanism	28
3.2 A Cross-Layer Resource Management Mechanism for Mobile Broadband Networks	33
3.2.1 The Call Admission Control Algorithm	34
3.2.2 The Resource Adjustment Algorithm	42
3.2.3 Analysis	43
3.3 Mobility Management for Mobile Broadband Networks	50
3.4 Experimental Results	54



3.4.1	Evaluation of the Proposed Mobile Broadband Network	54
3.4.2	Effectiveness of the Proposed Cross-Layer Resource Management Scheme	59
4	Power Saving Class Management for Single MS in Mobile Broadband Networks	65
4.1	Motivation and Problem Definition	66
4.2	Fold-and-Demultiplex (FD) Method with QoS-Guaranteed Packet Scheduler	67
4.2.1	Creating Tentative PSCs	67
4.2.2	Folding PSCs into a State Series	69
4.2.3	Demultiplexing the State Series into PSCs	71
4.2.4	Including a PSC of Type I for Non-real-time Connections	74
4.2.5	QoS-Guaranteed Packet Scheduler for FD Method	75
4.3	Per-Flow Sleep Scheduling (PSS) Method	78
4.3.1	Determining T_i of each C_i	80
4.3.2	Scheduling $R_{i,j}$, T_i^L , and T_i^f of each C_i	81
4.4	Feasibility Study of the Scheduling Problem	84
4.5	Experimental Results	85
4.5.1	Performance Evaluation of FD Method	85
4.5.2	Performance Evaluation of PSS Method	90
5	Power Saving Class Management for Multiple MSs in Mobile Broadband Networks	95
5.1	Motivation and Problem Definition	95
5.2	Per-MS Sleep Scheduling (PMSS) Method	96
5.2.1	Determining T_i of each M_i	98
5.2.2	Scheduling $B_{i,j}$, T_i^L , and T_i^S of each M_i	98
5.2.3	Determining T_{basic}	101
5.3	Experimental Results	101
6	Conclusions and Future Directions	104
	Bibliography	107
	Curriculum Vitae	114
	Publication List	115

List of Figures

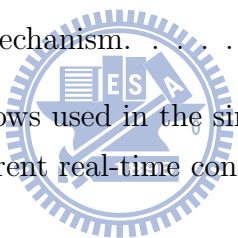
2.1	Mobile IP architecture	6
2.2	Indirect route for a mobile station	7
2.3	Host mobility vs. network mobility	7
2.4	An example of SIP call setup and tear-down.	9
2.5	An example of SIP with SDP message bodies: (a) INVITE signal and (b) OK signal.	10
2.6	The NAPT scheme.	11
2.7	IEEE 802.11 handoff procedure.	13
2.8	Structure of the IEEE 802.11e EDCA_Parameter_Set information element.	15
2.9	Structure of the TSPEC information element.	16
2.10	Definitions of PSCs in IEEE 802.16e.	16
2.11	Bandwidth allocation examples of UGS, rtPS, and ertPS in IEEE 802.16e.	17
3.1	System architecture.	20
3.2	The SIP registration procedure in our SIP-based mobile broadband network.	23
3.3	The message flow to set up a session.	25
3.4	The message flow of handoff.	27
3.5	Sleep procedure.	29
3.6	Wake-up process.	30
3.7	Session transfer process.	31
3.8	The proposed message flows of the CAC procedure in IEEE 802.11e networks.	35
3.9	The orders of information elements in (a) Probe Request and (b) Probe Response.	36
3.10	Basic operations of 802.11e EDCAF.	38
3.11	The bandwidth degrade algorithm.	39
3.12	An example of bandwidth degrade.	40
3.13	The RA algorithm.	42
3.14	The bandwidth upgrade algorithm.	43
3.15	An example of bandwidth upgrade.	44

3.16	The state transition diagram of a QSTA's rate change.	44
3.17	Generic state transitions under our analytical model.	45
3.18	A state transition example with $y = 2$ (q_1 and q_2 are the maximum numbers of calls that can be accommodated with PI_{max} at rates R_1 and R_2 , respectively).	46
3.19	System architecture.	51
3.20	Proposed Dynamic Tunnel Establishing procedure.	52
3.21	Proposed seamless handoff mechanism.	53
3.22	A testing environment of our SIP-based mobile broadband network system.	55
3.23	Uplink traffic load against the number of concurrent calls using one PHS/WCDMA/802.11b interface.	56
3.24	The handoff procedure when an IEEE 802.11 interface is used.	57
3.25	Comparison of simulation and analytical results on blocking rate and dropping rate ($\lambda_h = 0.8$, $\lambda_n = 1.2$).	60
3.26	Comparisons of different schemes on: (a) channel utilization and (b) goodput.	61
3.27	Comparison of different schemes on their average voice packet dropping rates (P_d).	62
3.28	Comparisons of: (a) call blocking rate and (b) call dropping rate.	62
3.29	The impact of P_r on: (a) call blocking rate and (b) call dropping rate.	63
3.30	The impact of the percentage of handoff calls on: (a) call blocking rate and (b) call dropping rate.	63
3.31	Impact of interference from AC_BE traffics with various AIFSNs for AC_BE traffics.	64
4.1	Sleep scheduling for two connections C_1 and C_2 using (a) one PSC and (b) two PSCs.	67
4.2	A mismatch example between the packet arrival time of a flow and its wake-up period.	68
4.3	Example of the state series construction.	71
4.4	Example of the PSC demultiplexing procedure.	73
4.5	Inserting a waiting interval Δt such that after P^I reaching $T'_{S,max}$, the listening windows of P^I will align with the listening windows of PSCs in C	75
4.6	Illustration of packet delay analysis.	76
4.7	Proof of Theorem 4.2.3. (a) the imaginary scheduling and (b) two changes to the imaginary scheduling by our earliest-next-bandwidth-first scheduling.	77
4.8	Output parameters T_i , T_i^L , T_i^f , and $R_{i,j}$ after the sleep scheduling of C_i and the system parameters π_k , $k = 1.. \frac{T_n}{T_{basic}}$, maintained by the central BS.	78
4.9	Examples of (a) Observation 4.3.1 and (b) Observation 4.3.2.	79

4.10	Example of scheduling $R_{i,j}$, T_i^L , and T_i^f of four connections in the MS. . .	82
4.11	Comparisons of PSS-DB and PSS-PI under different S_i (a) S_i , (b) $\frac{S_i}{2}$	83
4.12	Example of modeling a scheduling problem as a maximum matching problem.	85
4.13	r_s vs. B with (a) two real-time flow sets and (b) three real-time flow sets. .	86
4.14	r_s vs. number of real-time flow sets with (a) $B = 1250$ byte/frame and (b) $B = 2000$ byte/frame.	87
4.15	U vs. B with (a) two real-time flow sets and (b) three real-time flow sets. .	87
4.16	(a) Average delay vs. B and (b) jitter vs. B (two real-time flow sets). . . .	88
4.17	Effect of ρ on r_s and U under random flow arrival ($B = 1250$ byte/frame).	88
4.18	Effect of ρ on r_s and U under random flow arrival ($B = 2000$ byte/frame).	89
4.19	Effect of λ_n on (a) r_s and (b) response time with one real-time flow set and a non-real-time downlink flow of rate λ_n ($B = 1250$ byte/frame).	90
4.20	Effect of λ_n on (a) r_s and (b) response time with one real-time flow set and a non-real-time downlink flow of rate λ_n ($B = 2000$ byte/frame).	90
4.21	Effects of B on the power consumption, resource utilization, and drop rate under environments 1 ((a)~(c)) and 2 ((d)~(f)).	92
4.22	Effects of connection delay bound on (a) power consumption and (b) re- source utilization with packet inter-arrival time of type VI connections being 20 ms.	93
4.23	Effects of connection packet inter-arrival time and delay bound on (a) power consumption and (b) resource utilization.	94
5.1	Sleep scheduling for two MSs MS_1 and MS_2 using (a) the same common sleep cycle and (b) each with its own sleep cycle.	96
5.2	Example of the sleep parameters for the MSs and the data structure used in our proposed algorithm.	97
5.3	Example of scheduling $B_{i,j}$, T_i^L , and T_i^S of four MSs.	100
5.4	Effects of number of MSs on (a) active ratio and (b) fail-to-sleep probability.	102
5.5	Effects of maximum delay on active ratio.	102
5.6	Effects of the difference between minimum and maximum data rate on active ratio.	103
5.7	Effects of system bandwidth on (a) active ratio and (b) fail-to-sleep prob- ability.	103

List of Tables

3.1	Charge plans of different types of wireless networks.	21
3.2	The Packet Size Table, which contains the packet sizes (in bytes) when different codecs and packetization intervals are used.	37
3.3	Parameters of IEEE 802.11e EDCA.	38
3.4	The required MT of a bi-directional voice call under different physical rates under our analytical model.	44
3.5	Measurement of call setup time and capacities of different interfaces.	55
3.6	Measurement of handoff latencies.	57
3.7	Latencies of the push mechanism.	58
4.1	Six types of real-time flows used in the simulation.	86
4.2	QoS parameters of different real-time connections.	91



Chapter 1

Introduction

In the recent years, wireless broadband access is gaining more popularity. Among all wireless data access technologies, the IEEE 802.11 WLANs (wireless local area networks) [1, 4] provide an easy and low cost solution for users to build their own local area networks while the IEEE 802.16 WMANs (wireless metropolitan area networks) [25, 26] provide a wireless solution to substitute the wire line last-mile Internet access. For mobile wireless users, broadband access [16, 33, 22], QoS (quality of service) guaranteed [72, 68, 45, 11, 8] and continuous connectivity [71, 57, 46, 63, 18], and long device life time [62, 9, 61, 64, 24] are always three of the most important issues for wireless networks. In this dissertation, we study the power and resource management in mobile broadband networks, which is composed of three major issues: mobile and broadband access, resource management for mobile broadband networks, and power management for mobile broadband networks. For the first issue, we propose a mobile broadband network architecture which can provide a group of mobile users broadband wireless access by attaching multiple wireless interfaces on the mobile gateway. In the resource management issue, we first consider how to manage the wireless resource distribution for the mobile broadband networks. Then, we further study the problem by considering user handoff and rate adaptation. To solve the problem, we propose a cross-layer resource management mechanism by integrating the application layer information and the wireless MAC QoS mechanisms. At the same time, to maintain continuous connectivity for the QoS of real-time applications, we also study the seamless mobility problem in the resource management issue. Finally, in the last issue, we study the *power saving class (PSC)* management problem for IEEE 802.16e protocols in the mobile broadband networks.

In the first issue, we discuss how to provide broadband wireless access to a group of mobile users. Traditionally, wireless networks manage mobility and connectivity of mobile devices in an individual manner, called *host mobility* [43, 27, 52]. However, for a group

of mobile users who roam together, tracking users in an individual manner consumes lots of signaling cost, computation cost, and radio power consumption. Instead of the host mobility, a concept called *network mobility* [16, 33] is proposed recently. Network mobility packs all users in a group as a network unit and conducts mobility management through a gateway/router in the network. Supporting network mobility provides following advantages: 1) there is less power consumption for a mobile device to connect to the local gateway/router than to a base stations (BS) far away; 2) the complexity of mobility management is lower or even transparent for a user when connecting to a local gateway/router; and 3) there are fewer handoffs since the MAC layer handoff only occurs on the central gateway/router rather than on all mobile users and the mobility management of a single gateway/router can ensure the reachability of the whole group of users inside the gateway/router [42]. Also based on the network mobility concept, this work proposes to combine the innate mobility and scalability of *MANET* (*mobile ad hoc network*) with a *session initiation protocol* (*SIP*)-based mobile network architecture to support a group of mobile users networking services. To connect to the Internet, in the system, we introduce a *SIP-based Mobile Network Gateway* (*SIP-MNG*), which follows the SIP standard [52] and is compatible with the existing SIP framework, in each mobile network. No extra servers are needed in foreign networks except the existing SIP servers. In our design, the SIP-MNG connects to the intra MANET by some IEEE 802.11 interfaces configured at the ad hoc mode and attaches to the Internet through several external wireless interfaces (such as WiMAX, IEEE 802.11, GPRS, PHS, and 3/3.5G interfaces). Combining the bandwidths provided by these external interfaces, SIP-MNG is allowed to provide broadband bandwidths to the Internet.

For real-time and multimedia sessions, QoS always has to be guaranteed. So in the second issue, we investigate the resource management problem for the mobile broadband networks and propose two solutions. Recall that SIP is used in the mobile broadband network to manage network mobility. Therefore, we can estimate the required bandwidth of each connection by the *SDP* (*Session Description Protocol*) [20] carried by SIP signaling. So, in the first solution, we design resource management mechanism in the SIP-MNG by exploiting the session information carried by SIP signaling. By such a way, the QoS of real-time and multimedia applications can be directly guaranteed from the application layer. Then, considering rate adaptation and connection handoff can both vary the required resource in the mobile broadband network, we further design a resource management scheme by integrating SIP and the QoS mechanisms of wireless networks. It is known that typical multimedia applications can tolerate some degree of temporary bandwidth fluctuation with no or little perceived degradation in quality by using a rate-adaptive codec or hierarchical encoding [39, 11, 8]. For *VoIP* (*voice over IP*) applications, this

is also applicable by varying the codecs and PIs (packetization intervals). In this work, as an example, we focus on VoIP and QoS mechanisms in IEEE 802.11e [4] to design a cross-layer resource management mechanism to support handoff and multi-grade QoS for VoIP over multi-rate WLANs. (For other types of real-time applications and wireless network protocols, similar technology can be applied.) When handoff calls are accepted to an IEEE 802.11e *QAP* (*Quality of Service Access Point*) or when physical transmission rates of calls drop, we will see increasing competition among mobile stations (MSs) and thus decreasing bandwidth shares of existing calls. Our main idea in the proposed scheme is to change the codecs and/or PIs of some existing calls to degrade their QoS levels when resources are too stringent. We show that the overall capacity of a wireless network, the new call blocking rate, and the handoff call dropping rate can all benefit from such dynamics. We will also show that changing PIs of calls is more effective than changing codecs. This is at the cost of increased end-to-end latency for voice packets but is worthy in a very mobile environment. We will also adopt the ITU-T-recommended 150 ms as the bound for one-way end-to-end delay [28]. In the end of resource management work, we investigate how to reduce handoff latency to provide seamless and continuous network connectivity for mobile users and mobile broadband networks. Experiments show that in an IEEE 802.11 wireless network with 802.1x [5] wireless security, a traditional handoff takes 1-2 sec for intra-subnet handoff and 2-4 sec for inter-subnet handoff [6], which is intolerable for real-time applications. So we present a seamless post-handoff mechanism for wireless networks with 802.1x wireless security, which is dedicated to the DHCP-based IP networks but can be easily tailored to Mobile IP-supported networks. Moreover, this seamless post-handoff mechanism is extensible for all kinds of wireless networks with 802.1x wireless security.

In the last issue, we investigate the PSC management issue for IEEE 802.16e standard in the mobile broadband networks. This issue happens to the mobile broadband networks when the mobile gateway accesses the Internet by IEEE 802.16e interfaces or the mobile gateway acts as an IEEE 802.16e relay and the mobile users are equipped with IEEE 802.16e interfaces to access the Internet via the gateway. In wireless networks, power saving is always an important issue. For IEEE 802.11, intensive works have been devoted to the power saving issues [60, 32, 34, 70, 19]. However, for IEEE 802.16e, related studies are limited. So, we skip the IEEE 802.11 and put our attention on the IEEE 802.16e. In IEEE 802.16e, three types of PSCs are defined to meet different traffic characteristics. Each PSC consists of a sequence of interleaved listening and sleep windows, and can support one or multiple traffic flows in an MS with similar characteristics. Type I is designed for non-real-time traffic flows; it has exponentially increasing sleep windows if no packet comes. Type II is designed for real-time traffic flows; it has a fixed size of sleep

windows. Type III is designed for multicast connections or management operations. An MS can turn off its radio interface when all its PSCs are in their sleep windows, but has to wake up when any PSC is in a listening window. However, from an MS's point of view, the standard does not define how multiple flows should be put into one PSC and how multiple PSCs of an MS should cooperate with each other for better energy efficiency. At the same time, it needs to answer how to determine the parameters of each PSC, such as start frame, listening window size, and sleep window size, and how to guarantee QoS of traffic flows when multiple PSCs coexist. In our work, we address both single BS-MS pair and single BS to multiple MSs at the same time. For single MS, we propose to use multiple PSCs to schedule the sleep of the MS such that the sleep schedule can more accurately capture the QoS of flows and make the MS sleep more. Based on the same motivation, for multiple MSs, we also propose to schedule each MS's sleep according to each of their QoS characteristics. These lead to less energy consumption, more efficient use of bandwidth, and more compact listening windows.

The rest of this dissertation is organized as follows. In Chapter 2, we overview mobility management protocols, QoS mechanisms in wireless networks, and IEEE 802.16e sleep mode. In Chapter 3, we first present the proposed mobile broadband network and then investigate the resource management issue of the mobile broadband networks. In Chapter 4, we discuss the PSC management problem for an IEEE 802.16e BS-MS pair. In Chapter 5, the PSC management problem for multiple IEEE 802.16e MSs under a BS is investigated. We conclude the dissertation and give some future directions in Chapter 6.

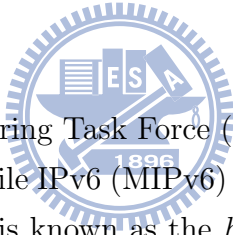
Chapter 2

Preliminaries

In this chapter, we give overviews of wireless mobility management protocols, QoS mechanisms in wireless networks, and IEEE 802.16e sleep mode.

2.1 Mobility Management Protocols

Mobile IP



In recent years, Internet Engineering Task Force (IETF) has developed protocols such as Mobile IPv4 (MIP) [43] and Mobile IPv6 (MIPv6) [27] to support continuous connectivity for mobile stations (MSs). This is known as the *host mobility*. To be transparent to the network handoff in MIP, an MS keeps its address when it moves from one network to another. To maintain the reachability of the MS, a node called *Home Agent (HA)*, which is located in the MS's home network, will track the foreign network, where the MS visits. In each foreign network, there is a node called *Foreign Agent (FA)*, which periodically broadcasts advertisement message in its network. Each FA has a unique address called *care-of address (COA)*, which can identify a foreign network, so when a roaming MS receives an advertisement message with new COA, it detects the network change and updates to its HA with the new COA. Consequently, the HA tunnels the packets destined for the MS to the new FA. Then, the FA can forward them to the inside MS. Fig. 2.1 shows the MIP system architecture proposed by IETF. The MS is initially in its home network and connects to a corresponding node (CN). Then, it moves to the visited network, which is managed by the FA with COA as 210.66.83.16. The MS will register to the FA and update its location to the HA. After this, all incoming calls will be forward to the visited network by the HA. Next let's see how the datagrams are addressed and forwarded to the MS in detail. As shown in Fig. 2.2, the HA first receives incoming datagrams which addressed to the MS from CN (1). It intercepts these datagrams and tunnels them to the

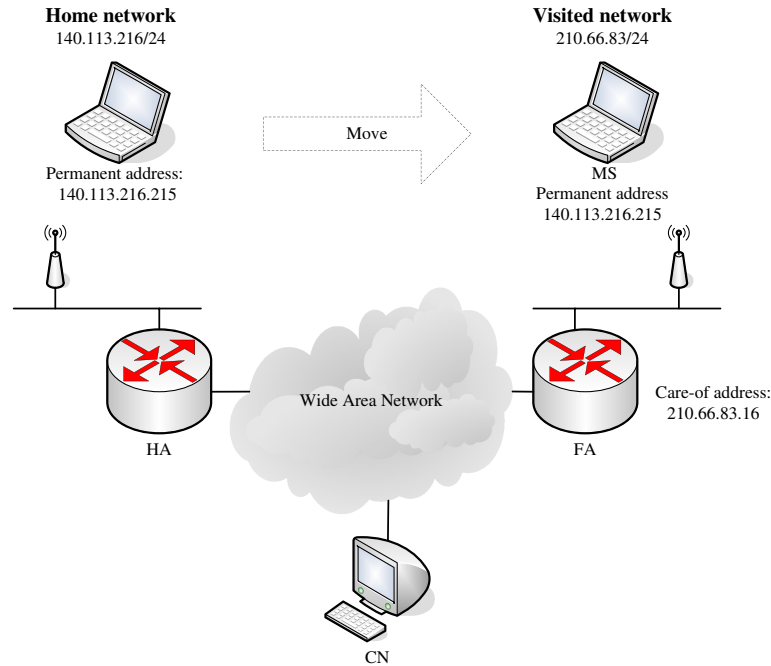


Figure 2.1: Mobile IP architecture

FA with source address as HA's address and destination address as FA's COA (2). On receiving these datagrams, the FA decapsulates them and forwards the inside datagrams, i.e., original datagrams from CN, to the MS (3). Following this procedure, MIP manages hosts' mobility and maintains their continuous connectivity.

Network Mobility

However, host mobility is insufficient due to two reasons. Firstly, not all devices can support such complex protocols, such as sensors on an aircraft. Secondly, once a device attaches to a MR (mobile router) in a mobile network, it cannot see any link-level handoff even as the mobile network moves. Therefore, the mobility management protocols need to be extended from host mobility to network mobility. Network mobility is namely a set of hosts that move collectively as a unit. This scenario can usually be seen on buses, ships, and aircrafts. There are tens/hundreds of passengers in a transportation carriage and move together. Fig. 2.3 illustrates the difference between host mobility and network mobility. In CarA, even though the inside devices move together, each handles its mobility by itself. In CarB, inside devices connect to the Internet through a MR. Since the MR and the inside devices move together, the latter does not need to pay cost for mobility management. All handoffs are handled by the MR. This also significantly reduces the handoff signaling cost.

The IETF has created a working group called Network Mobility (NEMO) [13, 14, 15]

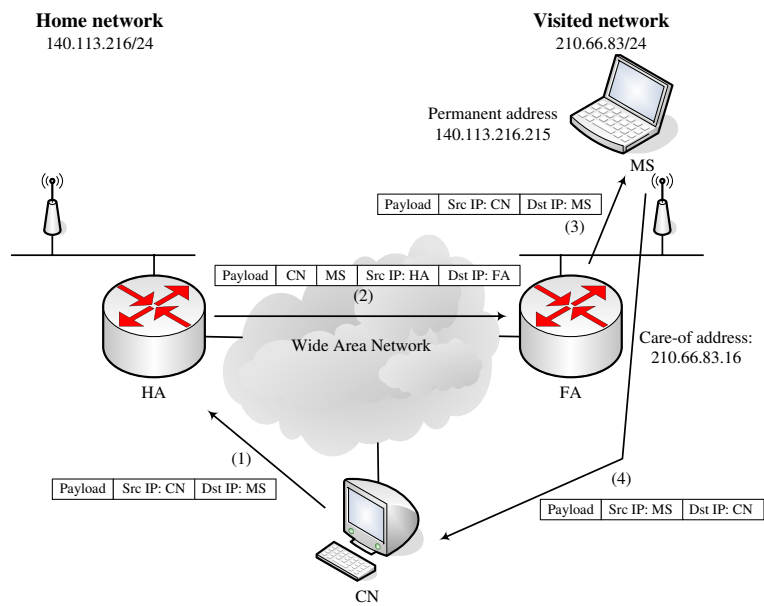


Figure 2.2: Indirect route for a mobile station

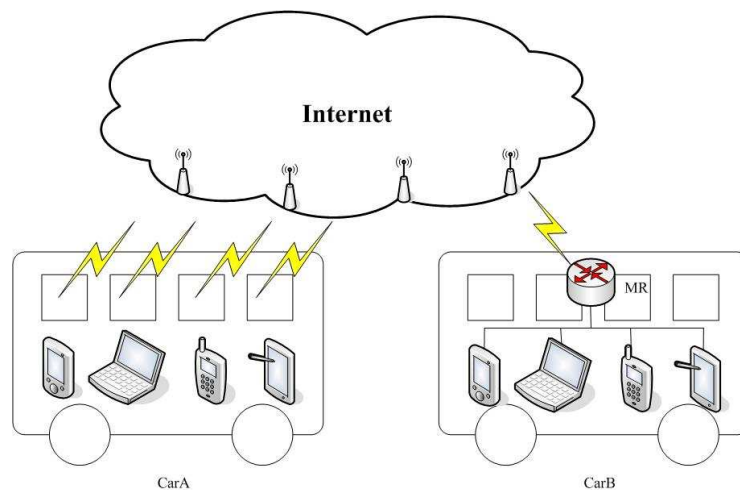


Figure 2.3: Host mobility vs. network mobility

that proposed a MIPv6-based NEMO (MIPv6-NEMO) protocol. MIPv6-NEMO allows session continuity for every station in the mobile network as the network moves. It also maintains the reachability of every station in the mobile network while moving around. A mobile network is a network segment or subnet that can move and attach to arbitrary points in the network infrastructure. It can only be accessed via the MR which manages the mobile network's mobility. MR and its HA run the MIPv6-NEMO protocol, which is a similar mechanism to the MIPv6 protocol. The MR takes care of all the stations within the mobile network irrespective of their capabilities. Following, we will briefly introduce the MIPv6-NEMO protocol.

Firstly, the MR is assigned to one or more mobile network prefixes by its HA. The MR has to update its latest location to its HA. Once the HA intercepts packets addressed to the mobile network prefixes, since the latest location information of the MR is maintained, it will tunnel them to the MR (by its care-of address). On the receipt of these packets, the MR decapsulates and forwards them to the inside MSs. Packets in the reverse direction are also tunneled via the HA in order to overcome ingress filtering restrictions [17]. In this case, the HA decapsulates the packets and forwards them to the CNs. This tunneling of packets is very similar to MIP and MIPv6. However, NEMO differs from them in that the MR updates the HA with the location of the entire mobile network, not just itself.

SIP

SIP (Session Initiation Protocol) [52] is a protocol for establishing an IP multimedia session. It's an application-layer control protocol to setup, modify, and terminate multimedia sessions. In [49], SIP has also been extended to provide presence, event notification, and instant message services. While SIP is not used to transport media traffic, it often chooses *RTP (Real-time Transport Protocol)* [55] as its transportation protocol and uses *SDP (Session Description Protocol)* [20] to specify its session characteristics. SIP endpoints are addressed by *SIP URLs (Uniform Resource Locators)*, which have the form of email address, for example, 7002@csie.nctu.edu.tw. SIP defines several logical entities, such as *user agent (UA)*, *redirect server*, *proxy server* and *registrar*. A UA originates and terminates requests. A redirect server is responsible to respond the requesters where they should sent their request messages to. A proxy server is to route SIP messages. A Registrar is a database which records the location(s) of a UA. Typically, a proxy or redirect server is implemented with a built-in registrar, and we call it a SIP server.

Fig. 2.4 shows an example of call setup and tear-down in SIP. When the caller, with SIP URL George@station1.nctu.edu.tw, wants to set up a call with the callee, with SIP URL Mary@station2.nctu.edu.tw, it sends the callee an INVITE request which includes session information in a SDP message body such as its supported codecs, received ip

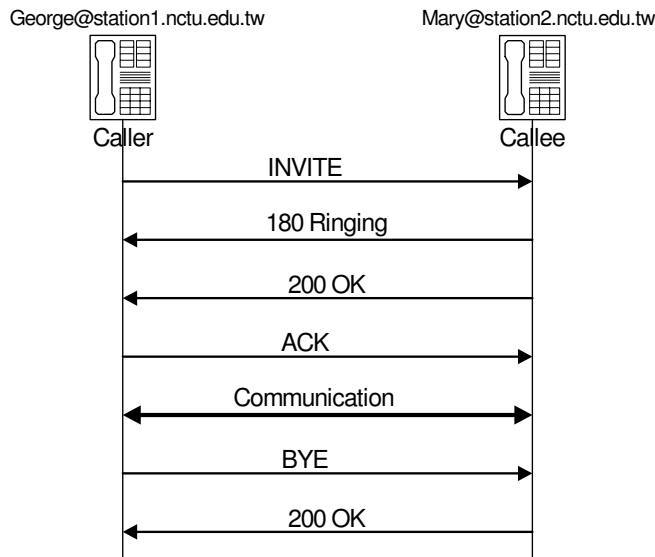


Figure 2.4: An example of SIP call setup and tear-down.

address and port number. Fig. 2.5(a) shows an example, where G.726 (format 2), and G.723 (format 4) are the offered codecs, with 4400 as its receiving port. If the callee decides to accept the request, it replies a 180 Ringing and a 200 OK responses. The 200 OK response will contain the final codec chosen by the callee. In Fig. 2.5(b), the selected codec is G.723, and the receiving port of the callee is 888. If the port number is 0, it means a rejection. On receiving the 200 OK response, the caller will response an ACK message to the callee. Then the call is established.

In addition to the MIPv6-NEMO, reference [22] proposes SIP-based NEMO (SIP-NEMO) to support network mobility. In SIP-NEMO, central to the mobile network is a SIP Network Mobility Server (SIPNMS), which is responsible for the mobility management of the mobile network. Each SIP-NMS has a SIP Home Server (SIP-HS), which records the current location of the SIPNMS and forwards requests to the SIP-NMS. By using SIP as the mobility management protocol, SIP clients can communicate with each other directly without tunneling.

NAT Traversal Problem for SIP

NAT is defined by RFC 3022 [59]. It is being used by many service providers and private individuals as a way to solve the problem of lacking public IP addresses. NAT solves this problem by mapping internal addresses to limited external or public addresses. There are three mapping technologies, *static NAT*, *dynamic NAT*, and *Network Address Port Translation (NAPT)*. Both of static NAT and dynamic NAT use the one-to-one address mapping. For static NAT, a private IP (IP_a) always maps to a fixed public IP (IP_x). In contrast with the static NAT, the dynamic NAT only generates the mapping when

```

INVITE sip: Mary@station2.nctu.edu.tw SIP/2.0
From: Caller < sip: George@station1.nctu.edu.tw >; tag=abc123
To: Callee < sip: Mary@station2.nctu.edu.tw >
CSeq: 1 INVITE
Content-Type: application/sdp
Content-Disposition: session

v=0
o= George 123 001 IN IP4 station1.nctu.edu.tw
s=
c=IN IP4 station1.nctu.edu.tw
t=0 0
m=audio 4400 RTP/AVP 2 4
a=rtpmap 2 G726-32/8000
a=rtpmap 4 G723/8000

```

(a)

```

SIP/2.0 200 OK
From: Caller < sip: George@station1.nctu.edu.tw >; tag=abc123
To: Callee < sip: Mary@station2.nctu.edu.tw >
CSeq: 1 INVITE
Content-Type: application/sdp
Content-Disposition: session

v=0
o= callee 456 001 IN IP4 station2.nctu.edu.tw
s=
c=IN IP4 station2.nctu.edu.tw
t=0 0
m=audio 888 RTP/AVP 4
a=rtpmap 4 G723/8000

```

(b)

Figure 2.5: An example of SIP with SDP message bodies: (a) INVITE signal and (b) OK signal.

necessary. In this scheme, a router with dynamic NAT will maintain an external IP pool. A free external IP is mapped to an inside private IP only when the latter wants to communicate to an external node. So, the dynamic NAT is a suitable scheme when the number of inside PC is greater than the number of public IPs. In NAPT, many private hosts can share single public address at the same time. For a router with NAPT, it identifies each session using different port number. One port is used for a translation between a private IP address and port number. Fig. 2.6 illustrates the NAPT scheme. The NAT router's external IP is 202.123.211.25. The inside IP address and port number 172.16.71.120:80 is mapped to the external port number 10080 while 172.16.71.121:80 is mapped to the external port number 20080. It's useful when a router/gateway connects to the Internet by single DSL connection and is only assigned one external IP address. In most of the cases, NAPT scheme is used in NAT router.

However, SIP has some inherent problems with NAT traversal. The first problem is that the SIP contact header of an internal host is its private IP address, so a SIP request message from an external node can not reach the internal host. The second problem is that, even a session is initiated by an internal host, the SIP response messages still cannot reach the internal host by an private address in the via header. The third problem is that

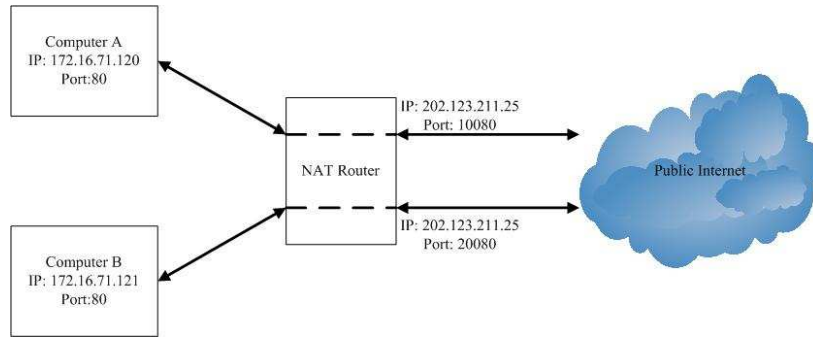


Figure 2.6: The NATP scheme.

the private media IP address and port number of an internal host in SDP message body is not known to the NAT router, so the incoming media packets can not be routed to the internal host by the NAT router. To solve these problems, an *Application Layer Gateway (ALG)* [51], which performs address and port translation in the application layer, for SIP is required. For outgoing SIP messages and the embedded SDP message body, ALG searches the internal host address in the application layer context and replaces them by the translated external address and port. On the contrary, for incoming SIP messages, ALG will translate the public address and port back to the destined private address and port and then forward these messages to the destined internal host. To solve the SIP signaling NAT traversal problem, we can choose to implement a SIP ALG in the application layer.

Handoff Procedure in IEEE 802.11

An ideal WLAN should provide successive radio signal coverage for MSs in its service area. Due to mobility, an MS may move out the coverage of its current AP. In this case, the received signal strength (RSS) and the signal-to-noise ratio (SNR) seen by the MS will degrade significantly and the MS will loss its connectivity with its current AP. Thus, the MS needs to initiate a handoff. The handoff procedure of IEEE 802.11 can be divided into three phases: *discovery*, *reauthentication*, and *reassociation*.

- a. **Discovery:** In order to find a new AP, an MS scans all channels either passively or actively. In *passive scanning*, the MS listens to APs' periodic beacon messages to know their existence and parameters, such as beacon interval, capability information, BSSID, supported rate, etc. The period of beacon frames is normally set to 100 ms in most implementations. In *active scanning*, for every channel, the MS broadcasts a probe request and the expects probe responses from APs. In most cases, MSs adopt active scanning rather than passive scanning to shorten their handoff latencies.

In the active scanning, the scanning delay can be calculated as:

$$N_{ch} \times T_b \leq t \leq N_{ch} \times T_t, \quad (2.1)$$

where N_{ch} is the total number of channels, (for example, $N_{ch} = 11$ for 802.11b/g), $T_b = \text{MinChannelTime}$ is the minimum time that an MS has to wait on a channel if the channel is idle (i.e., there is neither probe responses nor traffics in the channel), and $T_t = \text{MaxChannelTime}$ is the maximum time that a MS has to wait on a channel if there is any response. It is reported that, in most implementation, a full active scanning requires 300-400 ms for IEEE 802.11b/g [38, 29]. After scanning, the MS selects a new AP based on metrics such as Received Signal Strength Indication (RSSI) and loading information.

- b. **Reauthentication:** The new AP must authenticate the MS before the MS can re-associate with the new AP. The IEEE 802.11 standard defines two types of authentication: 1) Open System, which is a null authentication algorithm and 2) Shared Key, which is a four-way authentication mechanism. If the *Inter Access Point Protocol (IAPP)* [2] is used, only null authentication frames need to be exchanged in the reauthentication phase. Using IAPP, the new AP and the old AP can exchange context information of the MS and update the forwarding table at layer 2 devices. In this chapter, we assume that IAPP is adopted. In our experience, exchanging null authentication frames takes about 1-2 ms.
- c. **Reassociation:** The reassociation process involves exchanging reassociation request and reassociation response frames between an MS and its new AP. In an IEEE 802.11 with IAPP network, additional IAPP messages between the old and the new APs will be exchanged before the new AP replies a reassociation response to the MS. IAPP is proposed to reduce the opportunity of transmitting security information of MSs in the air during the handoff period to reduce the chance of the system being attacked. By IAPP, an AP is allowed to communicate with other APs to exchange relevant information of associated MSs on a common distribution system (DS).

An IEEE 802.11 with IAPP network typically comprises APs, MSs, and *Remote Authentication Dial In User Services (RADIUS)* servers [48]. The RADIUS servers provide two functions: 1) mapping of the BSSID of an AP to its IP address on the distribution system medium (DSM) and 2) distribution of keys to APs to allow secure communications between APs [2]. The message flow of a reassociation is shown as Fig. 2.7. When a new AP is located, the MS sends a reassociation request frame to the new AP. The request frame contains the MS's MAC address and the

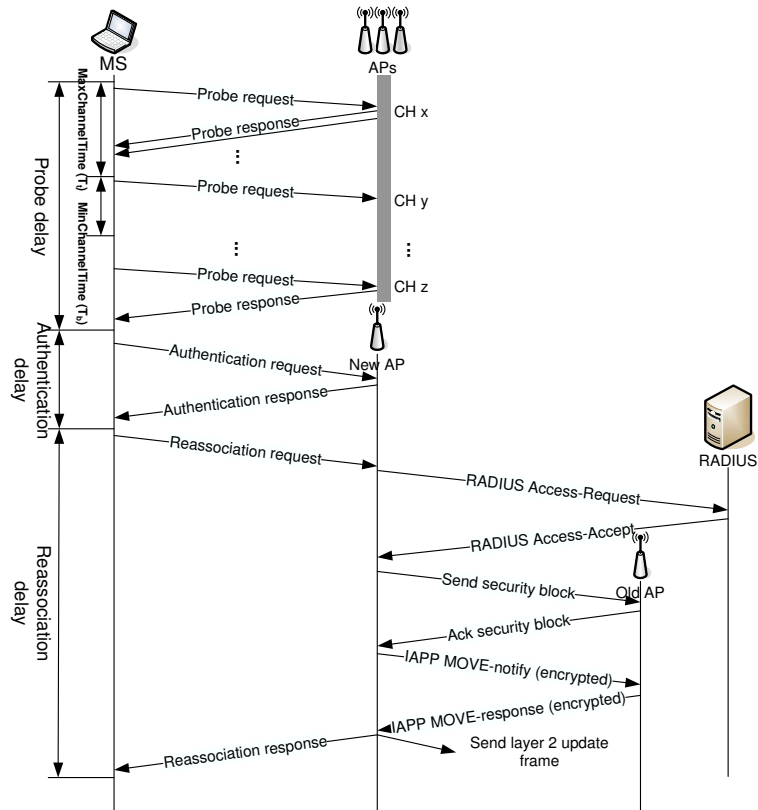


Figure 2.7: IEEE 802.11 handoff procedure.

BSSID of the old AP. Upon receiving the reassociation request frame, the new AP sends a RADIUS *Access-Request* packet to the RADIUS server to verify the old BSSID. If the old AP is valid, the RADIUS server will reply to the new AP a RADIUS *Access-Accept* packet which contains a *security block* for communication between the old AP and the new AP. Then the new AP will send an encrypted IAPP *MOVE-notify* packet to the old AP, which will reply an encrypted IAPP *MOVE-response* packet with the context information pertaining to the MS. Upon receiving the IAPP *MOVE-response* packet, the new AP will broadcast a layer-2 update frame to the DS to inform all layer-2 devices to update their forwarding information about this MS. Finally, the new AP will send the MS a reassociation response frame. This completes the handoff procedure. To conclude, IAPP can avoid transferring the MS's security information in the air but does not reduce the handoff latency effectively. According to [6], the reassociation in IEEE 802.11 with IAPP requires about 40 ms.

Fig. 2.7 summarizes the handoff procedure. The overall latency is more than 300 ms (including a full scanning of 300–400 ms and IAPP overhead of 40 ms). Previous work [65] indicates that it is preferred to bound the handoff latency within 50 ms. Reference [7]

also shows that handoff delay significantly affects TCP performance because TCP cannot distinguish a handoff from a network congestion.

Inter-IP Subnet Handoff Procedure in a Secured Wireless Network

Handoff refers to an MS moving from one AP's coverage to another. Generally, when an MS detects that its RSS from its current AP has dropped below some certain threshold, it starts carrying out a wireless handoff. A wireless handoff is composed of 4 main phases: *Probe-and-Decision*, *Execution*, *DHCP (Dynamic Host Configuration Protocol)*, and *Upper Layer Adjustment*. In the Probe-and-Decision phase, the MS scans channels to find potential APs around it. After scanning, it will decide a target AP as its new AP according to some matrices such as RSS and loading. Then, the MS starts the Execution phase to attach to the target AP. In an IEEE 802.11i network [5], the Execution phase involves 3 steps: *reassociation*, *802.1x authentication*, and *four-way handshake*. During 802.1x authentication, the MS is authenticated to a backend authentication (AAA) server, such as RADIUS (Remote Authentication Dial-In User Service) server [48], via the new AP. If the authentication succeeds, both the MS and authentication server will derive the same PMK (Pairwise Master Key). The PMK is then transmitted to the new AP by the authentication server, which triggers the new AP to initiate the four-way handshake. In the four-way handshake, the MS and the new AP will derive several temporal keys, including a data encryption key and a data message integrity code (MIC) key, to protect data between the MS and the new AP. At this point, the link layer handoff is complete. That is, if the handoff occurs within the same IP subnet (an intra-subnet handoff), the handoff is finished after the Probe-and-Decision and Execution phases. The DHCP and Upper Layer Adjustment phases are needed when an MS moves from one IP subnet to another (an inter-subnet handoff). In this case, after the link layer handoff, the MS has to renew its IP address and reconfigure its network parameters with the DHCP server of the new IP subnet. Afterward, the MS executes the Upper Layer Adjustment phase to adjust its TCP/IP layer or applications in order to continue its original sessions. This completes the inter-subnet handoff.

Each phase mentioned above causes considerable delay. Experiments show latencies of 300-400 ms for the Probe-and-Decision phase, 800 ms for the 802.1x authentication, 40 ms for the four-way handshake, and 1-2 sec for the DHCP procedure [6]. Obviously, the total handoff delay is intolerable for real-time applications.

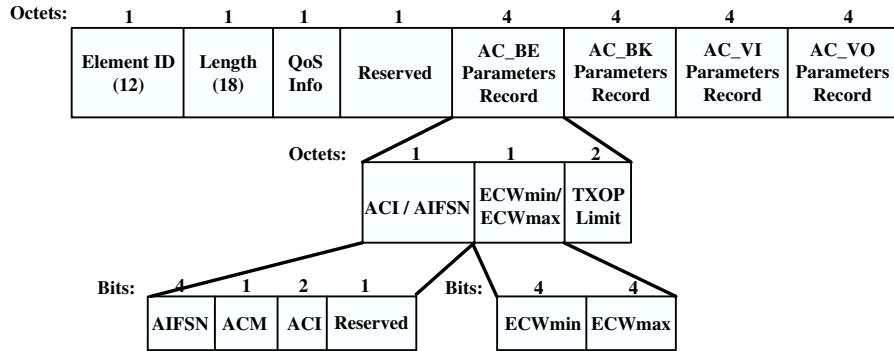


Figure 2.8: Structure of the IEEE 802.11e EDCA_Parameter_Set information element.

2.2 QoS Mechanisms in Wireless Networks

IEEE 802.11e QoS MAC

The IEEE 802.11e Working Group defines a supplement to the existing legacy 802.11 MAC sublayer to support QoS. It introduces a new *HCF* (*Hybrid Coordination Function*), which includes two access mechanisms, *EDCA* (*Enhanced Distributed Channel Access*) and *HCCA* (*HCF Controlled Channel Access*), corresponding to the existing *DCF* (*Distribute Coordination Function*) and *PCF* (*Point Coordination Function*), respectively, in 802.11. Two new features, *AC* (*Access Category*) and *TXOP* (*Transmission Opportunity*), are introduced in HCF. A TXOP is a bounded time interval during which a *QSTA* (*Quality of Service Station*) can hold the medium and transmit multiple frames consecutively with SIFS (Short interframe spacing) spacing. A station can obtain a TXOP by either contention or scheduled access assigned by polling messages.

EDCA of IEEE 802.11e

To differentiate services, the eight user priorities in 802.1D are mapped to four IEEE 802.11e ACs. Each AC has its own transmit queue with an independent EDCA function to contend the medium. These four ACs are background (AC_BK), best effort (AC_BE), video (AC_VI), and voice (AC_VO), and are prioritized by different *AIFS* (*Arbitration Inter-Frame Space*) and contention window sizes. If a collision occurs among ACs within a QSTA, the highest priority AC wins the contention and the other AC(s) will backoff. The EDCA_Parameter_Set information elements (Fig. 2.8), which are sent in beacon frames, specify the parameters of ACs.

Octets:	1	1	3	2	2	4	4	4	4
	Element ID (13)	Length (55)	TS Info	Nominal MSDU Size	Maximum MSDU Size	Minimum Service Interval	Maximum Service Interval	Inactivity Interval	Suspension Interval
	4	4	4	4	4	4	4	2	2
	Service Start Time	Minimum Data Rate	Mean Data Rate	Peak Data Rate	Maximum Burst Size	Delay Bound	Minimum PHY Rate	Surplus Bandwidth Allowance	Medium Time

Figure 2.9: Structure of the TSPEC information element.

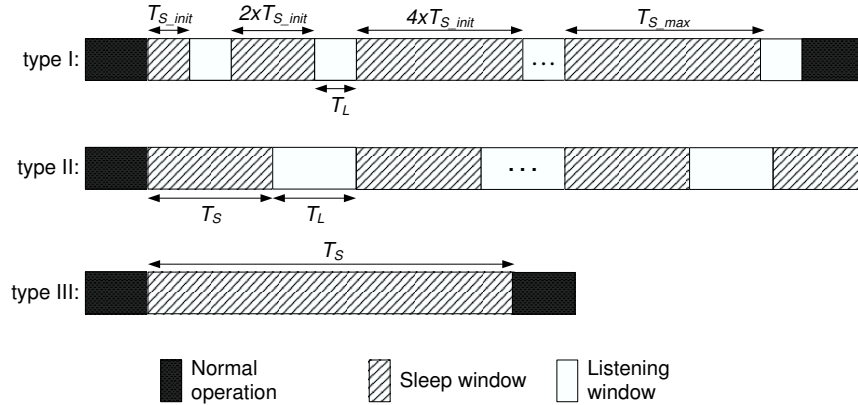


Figure 2.10: Definitions of PSCs in IEEE 802.16e.

Admission Control in EDCA

IEEE 802.11e allows a QSTA to request to add a new traffic stream by sending an *ADDTTS Request* to its *QAP (Quality of Service Access Point)*. The information carried in an *ADDTTS Request* includes the direction of the stream and a TSPEC (*Traffic Specification*) information element (Fig. 2.9). Admission control is conducted by the QAP by calculating the needed MT (*Medium Time*) as opposed to its remaining MT. Then, an *ADDTTS Response* can be replied.

2.3 IEEE 802.16e Sleep Mode

In IEEE 802.16e [26] sleep mode, three types of *Power Saving Classes (PSCs)* are defined to meet different traffic characteristics. Each PSC consists of a sequence of interleaved *listening* and *sleep* windows, and can support one or multiple traffic flows in an MS with similar characteristics. Type I is designed for BE (Best Effort) and NRT-VR (Non-Real-Time Variable Rate) traffic flows; it has exponentially increasing sleep windows if no packet comes. Type II is designed for UGS (Unsolicited Grant Service), RT-VR (Real-Time Variable Rate), and ERT-VR (Extended-Real-Time Variable Rate) traffic flows; it has fixed size of sleep windows. Type III is designed for multicast connections or

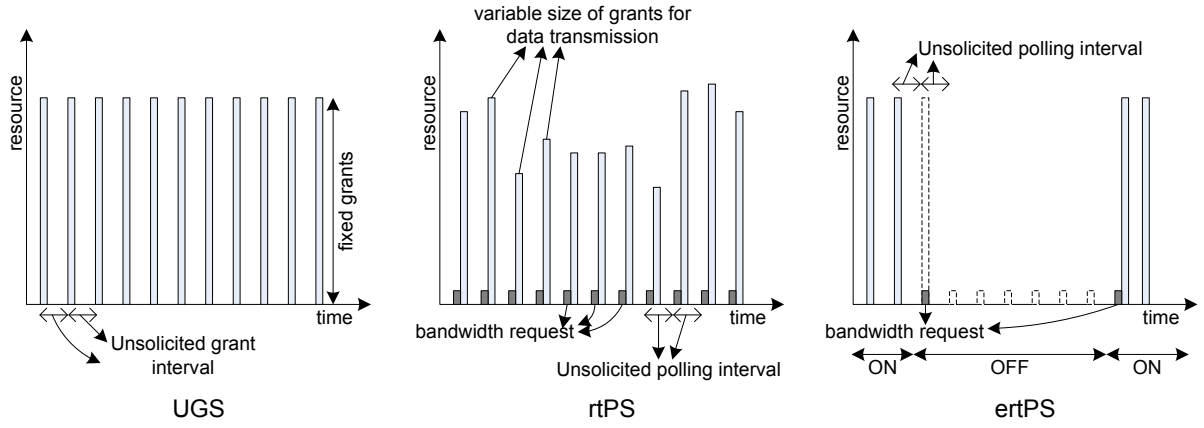


Figure 2.11: Bandwidth allocation examples of UGS, rtPS, and ertPS in IEEE 802.16e.

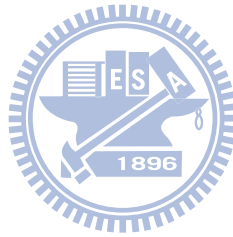
management operations. During a sleep window, the corresponding connections cannot send or receive packets. So, when all PSCs of an MS are in their sleep windows, the MS can turn off its air interface to save energy. This period is called an *unavailability interval* of the MS.

By Fig. 2.10, we illustrates the operation of three types of PSCs in more detail. A type I PSC's sleep windows are interleaved by fixed-length listening windows of size T_L . Its initial sleep window size is T_{S_init} , and is doubled each time, until reaching the maximum size, T_{S_max} , after which it remains the same. During a listening window, the MS will check if there are incoming packets for the type I PSC. If not, P^I will enter another sleep window; otherwise, it will deactivate the PSC and return to *normal operation*. For a PSC of type II, both its sleep windows and listening windows are of fixed lengths T_S and T_L , respectively. However, if there is any transmission/reception during a listening window, it will not return to normal operation unless being instructed. A PSC of type III has only one sleep window, after which it will return to normal operation immediately. The unit of these window sizes is the frame length, which is normally 5 ms [30]. A frame can be divided into a downlink subframe and an uplink subframe.

Uplink Scheduling Schemes in IEEE 802.16e

The BS needs to allocate bandwidths to flows according to their service types. For BE and NRT-VR, this is relatively easier because they have no real-time constraints. For UGS, RT-VR, and ERT-VR, we review three uplink scheduling schemes in the standard. UGS is designed for real-time services with periodical fixed-size packets, such as VoIP. So periodical fixed-size grants will be allocated to a UGS connection. For RT-VR, such as MPEG videos, *rtPS* (*real-time Polling Service*) can be used, where periodical fixed uplink resources are allocated to a connection. This allows the connection to ask for more resources when it has burst traffics. For ERT-VR, a connection may have periodical fixed-size packets interleaved by intermittent silence, such as VoIP connections with silence

suppression. Then *ertPS* (*extended real-time Polling Service*) can be used. Initially, a connection has periodical fixed-size grants. Once it has nothing to send, it will inform the BS to reduce each grant size to the minimal one. When the connection becomes active again, it can inform the BS to restore its original allocation. These are controlled by the QoS parameters: Minimum Reserved Traffic Rate (MRTR), Maximum Sustained Traffic Rate (MSTR), Maximum Latency, Unsolicited Grant Interval (UGI), and Unsolicited Polling Interval (UPI). Fig. 2.11 illustrates some examples.



Chapter 3

Resource Management Mechanisms for Mobile Broadband Networks

Broadband access and QoS guaranteed continuous connectivity are always important issues in wireless networks. Focusing on a group of moving mobile users, we combine the innate mobility and scalability of *MANET* (*mobile ad hoc network*) with a *session initiation protocol (SIP)*-based mobile network architecture to propose a SIP-based mobile broadband network, where the MANET is connected to the Internet via a SIP-based mobile network gateway, which is equipped with multiple external wireless interfaces and some internal IEEE 802.11 interfaces. Such an architecture can provide users broadband wireless access and is suitable for a group of tens/hundreds of users who roam together, such as in a car, bus, train, or for a travel group. Our system only needs an additional *SIP-based Mobile Network Gateway (SIP-MNG)*, which follows the SIP standard and is compatible with the existing SIP framework, in each mobile network. No extra servers are needed in foreign networks. Then, to maintain the QoS and continuous connectivity of mobile users and the mobile broadband networks, we propose two resource management mechanisms and a seamless handoff mechanism. In the first resource management mechanism, we exploit the session information carried in SIP signaling to design the scheme in the application layer. In the second resource management mechanism, we further consider the user handoff and physical rate adaptation to design a cross-layer resource management scheme, which integrate the session information and signaling in the application layer and the QoS mechanism supported by the MAC layer. Here, as an example, we focus on VoIP and QoS mechanisms in IEEE 802.11e to design the mechanism to support multi-grade QoS for VoIP over multi-rate WLANs. (For other types of real-time applications and wireless network protocols, similar technology can be applied.) To maintain continuous network connectivity of MSs for QoS guarantee, we study the mobility management prob-

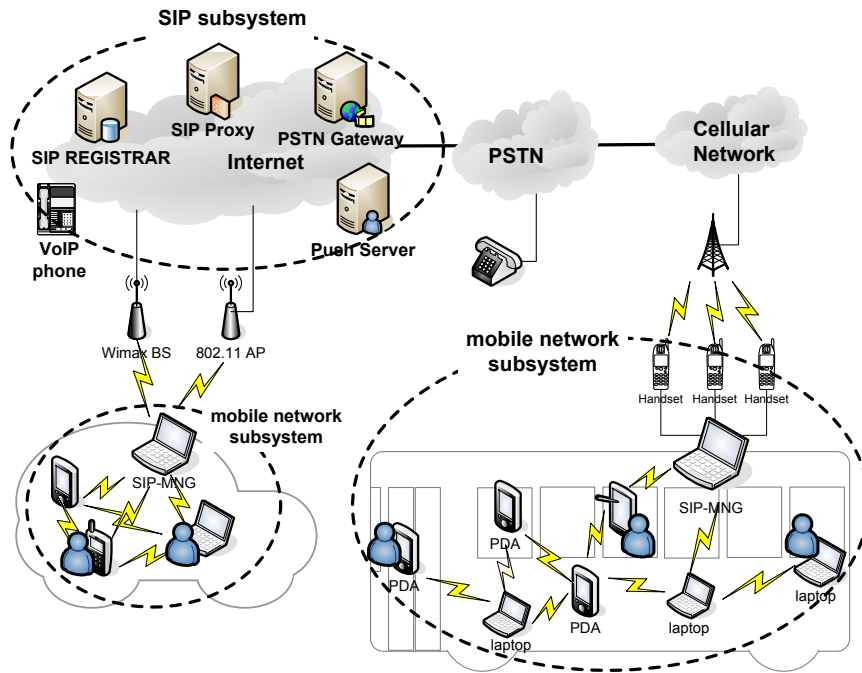


Figure 3.1: System architecture.

lem as the last issue of this chapter and present a seamless post-handoff mechanism for wireless networks with IEEE 802.1x security.

3.1 Design of a Mobile Broadband Network and Its Resource Management Mechanism

3.1.1 System Architecture and Motivation

Fig. 3.1 shows our SIP-based mobile broadband network architecture, which contains a *mobile network subsystem* and a *SIP subsystem*. The former is a SIP-based mobile network connecting to the Internet. The latter includes some servers to support SIP-based networking services.

Central to the mobile network subsystem is a SIP-based mobile network gateway (SIP-MNG). It is equipped with one or multiple wireless interfaces (such as GSM, GPRS, PHS, 3G, WLAN, and WiMAX interfaces) that can dial up to the Internet and some IEEE 802.11 interfaces to connect to the internal MANET. In real applications, cellular interfaces may be applied in freeways, country areas, and trains, while WLAN and WiMAX interfaces may be applied in hot-spot and metropolitan areas. The MANET consists of a set of mobile stations (MSs), each equipped with an 802.11 interface configured at the ad hoc mode. Because real-time services are stringent in responsiveness, routing in the MANET is supported by a proactive protocol, such as DSDV [44] and CGSR [10], which

Table 3.1: Charge plans of different types of wireless networks.

Type of Interfaces	Charge plans		
	By time	Flat rate	By packet
GPRS			✓
3G		✓	✓
PHS	✓	✓	
IEEE 802.11	✓	✓	
WiMAX	✓	✓	

will attempt to maintain consistent, up-to-date routing information at each host.

The SIP subsystem has four components: SIP registrar, SIP proxy, PSTN (Public Switched Telephone Network) gateway, and push server. The SIP registrar is a database containing users' subscription and status information. Users need to send their SIP registrar SIP REGISTER request messages to update their current locations periodically or when IP changes. The SIP proxy is responsible for routing SIP messages. The SIP registrar and SIP proxy are logical entities and are commonly implemented in a single host, called a *SIP server*. The PSTN gateway interconnects the Internet and the PSTN. So, with the PSTN gateway, SIP servers can set up/receive calls to/from the PSTN, respectively. The push server is to support our push mechanism and can "wake up" the SIP-MNG when its wireless interfaces are disconnected from the Internet. This can prevent incoming requests to the mobile network subsystem from loss. The detail of the push mechanism will be illustrated in the next chapter.

Before presenting the detail system operations of our system, we first introduce our design motivations.

- **Broadband wireless access:** By attaching multiple wireless interfaces on the SIP-MNG, the proposed mobile broadband network can provide the mobile users inside the network broadband access.
- **Saving charges of Internet access:** Table 3.1 shows the charge plans of different types of wireless interfaces. There are 3 types: 1) charge by time, 2) flat rate, and 3) charge by packet. Clearly, for charge plans 1) and 2), accessing the Internet for a group of users in a vehicle through a few wireless interfaces can save a great deal of charges for individuals. For plan 3), operators can save wireless resources.
- **QoS guarantee:** For real-time and multimedia sessions, QoS has to be guaranteed. Exploiting session information carried by SIP signaling, our SIP-MNG is designed with CAC (call admission control) and RM (resource management) mechanisms to guarantee the QoS of real-time or multimedia applications.

- **Push mechanism to save charges, power consumption, and wireless resources:** Considering that the current battery technology for cell phones can operate 2~5 hours in the active mode, and 5~14 days in the standby mode, it is desirable to disconnect the cellular interfaces of SIP-MNG (by putting them to the standby mode) to save power consumption when no Internet activity exists (this is needed when the SIP-MNG is powered by batteries). Also, this can save charges for plans 1) and 3), too. Moreover, the SIP-MNG does not need to collect network advertisements to maintain global reachability. Via SIP session control and SMS (short message service), we design a push mechanism to allow the wireless interfaces to stay off-line when there is no Internet connection and to be “woken up” when necessary.
- **An added service for public transportation operators:** The public transportation operators can use our system as an added service to attract customers. In addition, adding management functions (such as accounting) becomes easy via SIP-MNG.
- **Backward compatibility:** Our goal is to serve existing SIP clients without modification. So we design our system by adding some new servers that can work transparent to existing SIP clients.
- **Reducing handoffs:** When a vehicle moves from one BS to another, BSs have to handle a large number of handoff events if host mobility is used. With our architecture, BSs (base stations) only have to handle the mobility of the SIP-MNG, thus significantly reducing BSs’ load.
- **Saving the power consumption of MSs:** Since an MS connects to the Internet via the local SIP-MNG, its power consumption is significantly reduced.
- **Decreasing the complexity of MSs:** Since handoff events are handled by the SIP-MNG, MSs do not have to do movement detection and location update. The design of MSs is simplified. This is also known as *movement transparency*.

3.1.2 Basic Operations of the Mobile Broadband Network

In this section, we discuss the basic network operations in our system. Since the MANET is considered a private network, to provide Internet access, the SIP-MNG serves as a NAT (Network Address Translation) server for MSs and is responsible for the translation of SIP messages. To achieve the NAT traversal for SIP, techniques such as ALG (Application

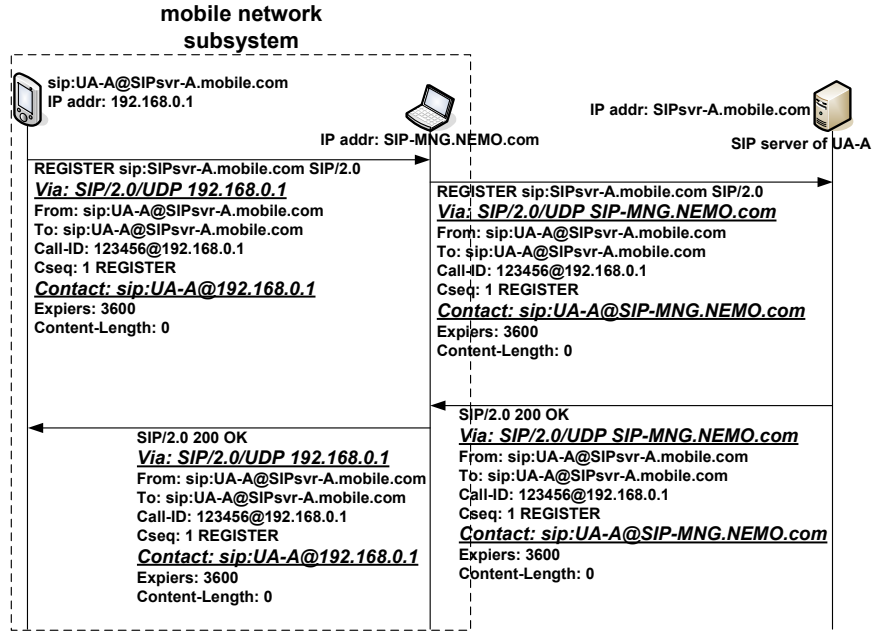


Figure 3.2: The SIP registration procedure in our SIP-based mobile broadband network.

Layer Gateway) [59], STUN (Simple Traversal of UDP Through NATs) [53], and ICE (Interactive Connectivity Establishment) [50] have been proposed. Here we adopt the ALG scheme in our SIP-MNG. Below, we will discuss the entrance and session establishment procedures and CAC, RM, and handoff mechanisms.

MS Joining the Mobile Network

When a mobile device moves into a vehicle with the proposed SIP-based mobile broadband networking services, its IEEE 802.11 interface will detect the existence of the network. After attaching to the mobile network, the MS will get a new IP address from the SIP-MNG. With this address, the MS can actively send a SIP REGISTER message to its SIP server to update its contact information. The REGISTER message will trigger the SIP-MNG to serve as a representative of the MS. Fig. 3.2 shows the detail procedure of the SIP registration. In this scenario, we assume the MS UA-A gets an IP address 192.168.0.1 from the SIP-MNG, and its SIP URL is sip:UA-A@SIPsvr-A.mobile.com. Addresses of the SIP-MNG and UA-A's SIP server are SIP-MNG.NEMO.com and SIPsvr-A.mobile.com, respectively. In the registration procedure, UA-A first sends a SIP REGISTER message to its SIP server with Via and Contact fields equal to 192.168.0.1 and sip:UA-A@192.168.0.1, respectively. Since the SIP-MNG monitors and translates all SIP messages to/from the Internet, it will capture the REGISTER message. On intercepting the message, the SIP-MNG will record UA-A's affiliation by adding UA-A's information into a *SIP_client_table* and relay the message by translating the Via and Contact fields into SIP-MNG.NEMO.com and sip:UA-A@SIP-MNG.NEMO.com, respectively. On receiving

the SIP REGISTER message, the UA-A's SIP server will update UA-A's information and then reply a SIP 200 OK message. Since the Via field in the REGISTER request is translated into the SIP-MNG's address, the 200 OK message will be forwarded to the SIP-MNG. Also, with the SIP-MNG's address as the contact address, UA-A's SIP server can later forward SIP request messages destined for UA-A to the SIP-MNG, which will then relay them to UA-A. Upon the receipt of the 200 OK message from UA-A's SIP server, the SIP-MNG will translate the Via and Contact fields back to UA-A's address and relay the SIP response to UA-A. Then, the SIP registration procedure is completed.

Session Setup Procedure and CAC and RM Mechanisms

In this subsection, we present how a session is established between an MS in the MANET and an external CN (Corresponding Node). We also discuss how our SIP-MNG supports CAC and RM. Recall that a SIP-MNG has one or more external interfaces. An interface is *active* if it has been dialed up to the Internet; otherwise, it is *idle*. RM is responsible for evaluating the required bandwidth of a requested session and assigning a serving external interface for the session. The required bandwidth can be estimated by the SDP (Session Description Protocol) [20] provided in the SIP signal. If any active wireless interface has sufficient spare resource, RM will assign it to the session. Otherwise, RM will activate a new wireless interface to serve the session. If all wireless interfaces are full, CAC will drop the SIP signal and reject the request.

Here we use a voice call request as an example to describe the detail message flow of session setup, CAC, and RM. Fig. 3.3 depicts the message flow, with some SIP headers omitted for simplicity. We assume that UA-A's IP address and SIP URL are 192.168.0.1 and sip:UA-A@SIPsvr.mobile.com, respectively. CN's IP address and SIP URL are 140.113.216.112 and sip:CN@SIPsvr.mobile.com, respectively. For simplicity, UA-A and CN use the same SIP server, whose address is SIPsvr.mobile.com. The SIP-MNG's address is SIP-MNG.NEMO.com. The corresponding call setup steps when UA-A calls CN are as follows.

1. UA-A sends a SIP INVITE to the SIP server to invite CN. Fields c (connection information) and m (media description) in the SDP are set to the connection address (192.168.0.1) and port number (9000), respectively (by which UA-A can receive data from CN). Field m also describes that the requested session is audio, so RM can tell that this is an real-time/multimedia session.
2. When intercepting the SIP INVITE message, the SIP-MNG will initiate the CAC and RM procedures. CAC will accept the requested session if RM returns ok, and

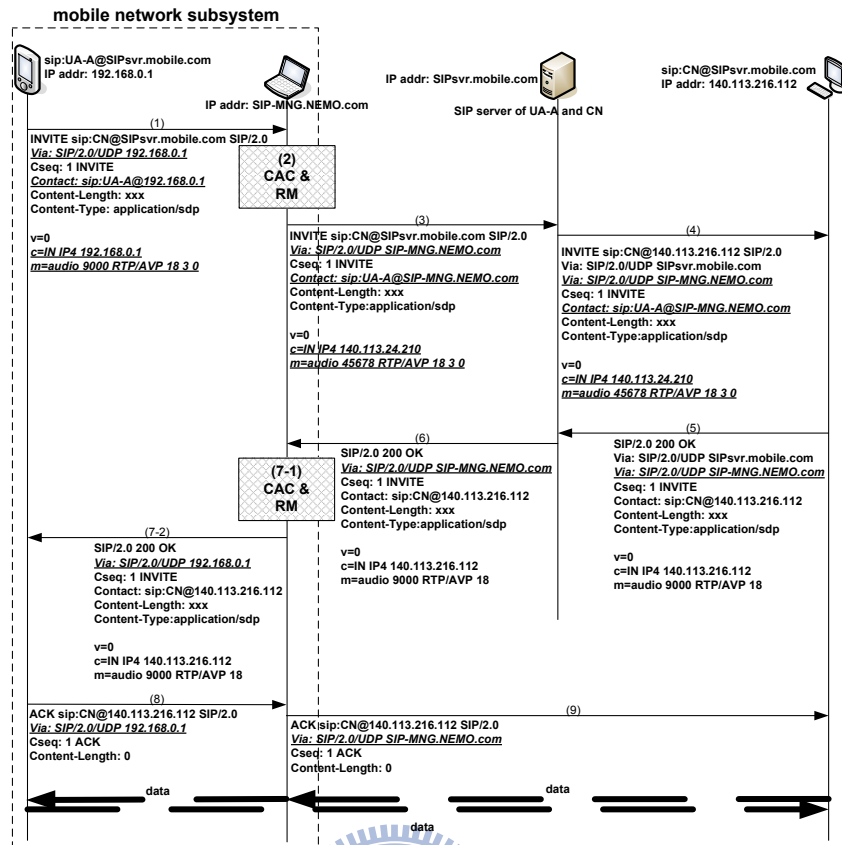
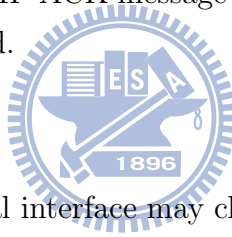


Figure 3.3: The message flow to set up a session.

reject the session otherwise. RM includes 3 steps: 1) evaluate the required bandwidth; 2) allocate a wireless interface for the session if there is enough bandwidth; and 3) return the status ok/failure to CAC. Since the codec information can be derived from field m (18 = G.729, 3 = GSM, and 0 = G.711 mu-law) and the default packetization interval (PI) is 20 ms, the required bandwidth can be predicted. If there is no resource, the SIP-MNG will drop the SIP INVITE message and respond a SIP "480 Temporarily not available" message to UA-A. Otherwise, the session will be recorded into a *session table* and the SIP INVITE will be relayed to the SIP server after translation. Since the SDP may provide multiple candidate codecs for the session, RM will reserve the maximum required bandwidth for the session. For non-audio/video sessions (which will be described in field m), the SIP-MNG may reserve one or few wireless interfaces dedicated to these types of sessions. For such best-effort sessions, RM will skip the resource evaluation step and directly allocate an interface for it.

3. Then fields Via, Contact, c, and m of the SIP INVITE message are translated. In this example, the (connection address):(port) is translated from 192.168.0.1:9000 to 140.113.24.210:45678.

4. On receiving the SIP INVITE message, the SIP server forwards it to the CN, which will register the caller's address as 140.113.216.112.
5. The CN then replies with a SIP 200 OK message, where it chooses G.729 as the codec to communicate with UA-A. According to the Via header in the received SIP INVITE message, this 200 OK message will be sent back to the SIP server at SIPsvr.mobile.com.
6. Upon the receipt of the 200 OK message, the SIP server forwards it to address SIP-MNG.NEMO.com.
7. When the SIP-MNG receives the 200 OK message, RM will update the session table according to the final media information. Then, the SIP-MNG translates the 200 OK message and relays it to UA-A.
8. UA-A feedbacks SIP ACK message to the CN directly according to the Contact information in the received 200 OK message.
9. The SIP-MNG relays the SIP ACK message to CN after translating the Via header. Then the call is established.



Handoff Procedure

As the vehicle moves, an external interface may change its network domain and the SIP-MNG may change its active interface. In both cases, a handoff happens. The SIP-MNG must recover sessions on these handoff interfaces by sending re-INVITE messages and maintain the global reachability of all MSs in the MANET by sending re-REGISTER messages. Since recovering on-going sessions is more urgent, it will be done first. Below, we outline the detail steps (refer to Fig. 3.4).

1. According to the SIP_client.table and the session table, the SIP-MNG retrieves the endpoint MSs and CNs of the on-going sessions on each handoff interface.
2. The SIP-MNG then sends each CN a SIP INVITE message to re-invite it.
3. A CN, on receiving the SIP INVITE message, will register the new contact address and port of the MS. Then the CN replies a SIP 200 OK message.
4. On receiving the 200 OK messages, the SIP-MNG will update the session table and reply the corresponding CN a SIP ACK message. Then, the session between the CN and the MS can be continued.

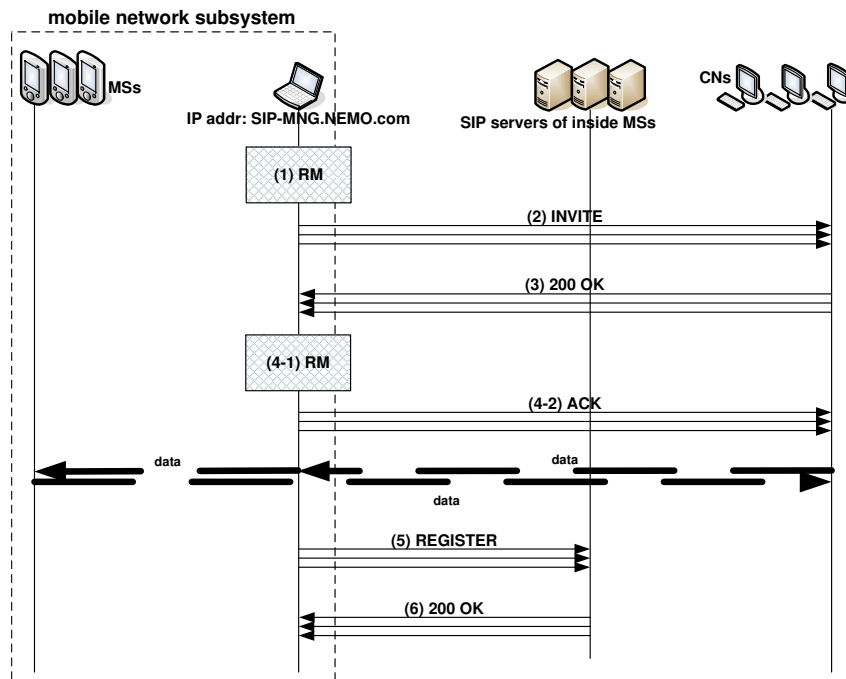


Figure 3.4: The message flow of handoff.

5. After recovering all handoff sessions, the SIP-MNG will send a SIP REGISTER message for each MS that has changed to a new external interface to update its contact address.
6. When a SIP server receives the above SIP REGISTER message, it will update the corresponding MS's data and reply a SIP 200 OK message. After this, all MSs are guaranteed to be reachable from outside.

MS Leaving the Mobile Network

When an MS leaves the mobile broadband network, it may detect other network and update its contact information by sending a SIP REGISTER message. If there is an on-going session, it can be resumed by SIP re-INVITE. However, since the MS does not deregister with the SIP-MNG, the SIP-MNG will keep its information in the SIP_client_table. If the MS has an on-going session before it leaves, the SIP-MNG will still reserve resource for the session. Fortunately, if a SIP request arrives, because the MS does not exist in the network, a new session will not be set up. However, the allocated resource will never be released. To solve this problem, we suggest to set a timer for each session and integrate the SIP-MNG with the underlying routing protocol in the MANET. If the SIP-MNG finds that the MS does not exist after the timer times out, the corresponding resource will be recycled. Noted that the SIP-MNG can determine whether a specific MS exists or not by searching the corresponding entry in the routing table. Alternatively, the SIP OPTIONS

message can be used to query the capability of a SIP client or server. The message should be responded with a SIP 200 OK response with the supported capabilities. This can be used to determine an MS's existence.

3.1.3 Proposed Push Mechanism

Our system allows the external interfaces of SIP-MNGs to be disconnected when there is no Internet connection. When any interface is connected, the mobile network becomes a part of the Internet, so all sessions can be handled as usual. When all interfaces are disconnected, if an outgoing SIP request is sent by a user in the mobile network, the SIP-MNG can buffer the invitation, dial up to the Internet, register this user with the SIP registrar, and then send the invitation to the callee. However, when the mobile network is out of reach from the Internet, an incoming SIP request can not be completed. To solve this problem, we propose a *push mechanism*.

In our push mechanism, the SIP-MNG will carry out a *sleep procedure* when it decides to disconnect from the Internet. Recall that there is a push server in the SIP subsystem (refer to Fig. 3.1). The SIP-MNG will solicit the push server as its agent during the disconnection period. When an incoming SIP request arrives, the push server will be notified first and will trigger a *wake-up procedure*, which includes a *wake-up process* and a *session transfer process*. In the wake-up process, the push server will activate the SIP-MNG via SMS (Short Message Service) and establish a connection with the caller to hold the session. After the SIP-MNG reconnects to the Internet, the transfer process will help build the link between the caller and the internal callee. In this way, the SIP-MNG can stay off-line but remain reachable from the Internet.

The push mechanism has also been used in other applications. In GPRS networks, an MS must activate a PDP context before the corresponding service can be used. However, maintaining a PDP context without actually using it will consume network resources. So, short messages are used in [36] to activate a PDP context on-the-fly. For a dual-mode handset (with a cellular and a WLAN interfaces, for instance), to reduce energy consumption, it is desirable to disable the handset's WLAN module when it is not used. However, this will prevent the handset from receiving incoming VoIP calls from the WLAN interface. To solve this problem, reference [35] uses a paging mechanism via the cellular interface to inform the handset to activate its WLAN module. Then the VoIP call can be connected via the relatively inexpensive WLAN. However, both [36, 35] involve some modifications on SIP components and end devices to support such functions, which is undesirable. In addition, because of the overhead to go through the push procedure, both approaches may suffer from the timeout problem if the callee's response time exceeds a predefined threshold. In our architecture, the standard protocols in SIP servers and SIP

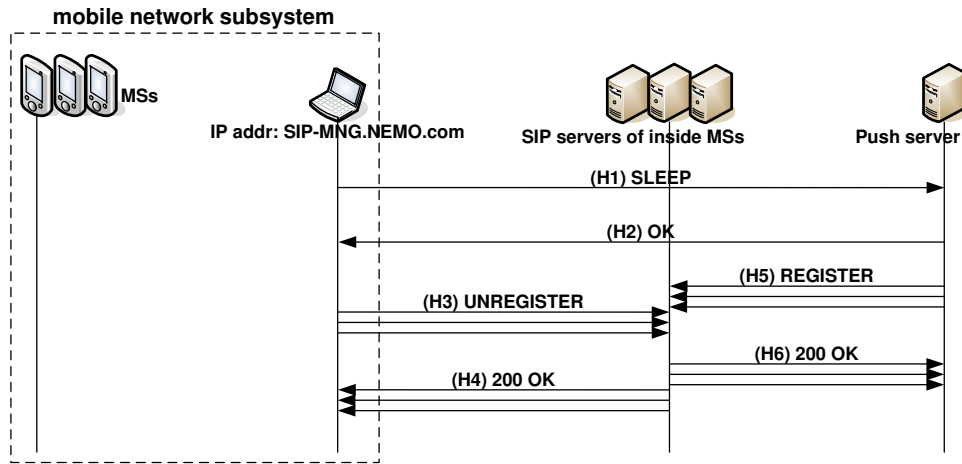


Figure 3.5: Sleep procedure.

clients are unchanged. Also, with an external server and SIP call control feature, our approach can effectively relieve the timeout problem.

Sleep Procedure

To avoid modifying the existing SIP standard, the push server works as an agent for users inside the mobile network when the SIP-MNG is disconnected from the Internet. The sleep procedure is to inform the SIP server to redirect all SIP messages destined to mobile network users to the push server.

Fig. 3.5 shows the message flow of the sleep procedure. When there is no Internet connection, the SIP-MNG may initiate the sleep procedure by sending a sleep request to the push server (H1). The sleep request includes the SIP URIs of MSs inside the mobile network and the MSISDN (Mobile Station International ISDN Number) of one of the cellular interfaces of the SIP-MNG. The MSISDN will later be used by the push server to notify the SIP-MNG of new incoming SIP requests via short messages.

The push server maintains a *gateway table* and a *SIP client table*. Each entry in the gateway table is to track the status of one SIP-MNG and includes four fields: gateway id, status, MSISDN, and IP address. Each entry in the SIP client table is to track one MS and includes three fields: SIP URI, SIP-MNG id, and registration expiration time. On receiving the sleep request from the SIP-MNG, the push server updates these two tables and marks the status of the SIP-MNG as off-line. The push server then replies an OK to the SIP-MNG (H2). Upon receipt of the OK message, the SIP-MNG will generate REGISTER messages for all internal SIP clients to their corresponding SIP servers (H3) with an EXPIRE value of 0 (which means unregister). In return, the SIP server will reply SIP 200 OK messages (H4). On the other hand, as soon as the push server replies an OK message to the SIP-MNG (H2), it also sends REGISTER messages for all MSs served by

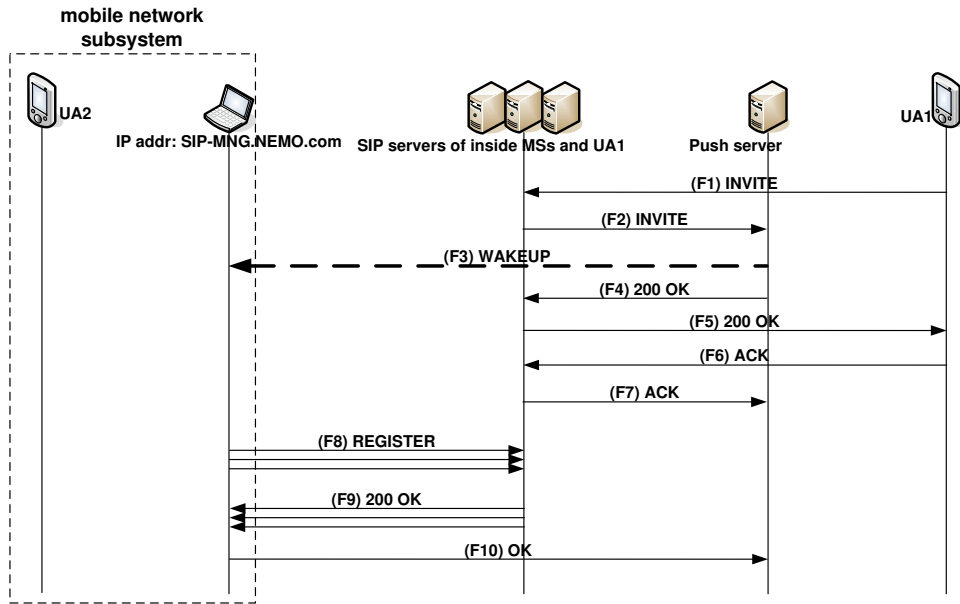


Figure 3.6: Wake-up process.

the sleeping SIP-MNG to their SIP servers with a non-zero EXPIRE value (H5). These REGISTER requests should contain the push server's IP address in the CONTACT field, so that all future SIP INVITE requests to these MSs will be forwarded to the push server. Upon receipt of a REGISTER message, a SIP server will update its record and reply a SIP 200 OK message to the push server (H6). After step H4, the SIP-MNG can disconnect all its wireless interfaces, and after step H6 the push server will become the agent of the mobile network subsystem and send periodic REGISTER messages for it.

Wake-Up Procedure

Consider a SIP request from a SIP client UA1 in the Internet to a client UA2 in the mobile network. To complete this session, there are two parts in the wake-up procedure: wake-up process and session transfer process.

- Wake-Up Process

Fig. 3.6 illustrates the message flow of the wake-up process. To set up a session to UA2, UA1 first sends a SIP INVITE message to UA2's SIP server containing the session information, connection address, and port number of UA1 in the SDP (F1). The SIP server will identify that the contact of UA2 is the push server and forward the INVITE to the push server (F2). The push server then checks its SIP client table and gateway table and retrieves UA2's SIP-MNG information. Because the status of the SIP-MNG is off-line, the push server will send a short message to the MSISDN registered by the SIP-MNG (F3). This short message carries the event type and IP address of the push server to inform the SIP-MNG to reconnect to the

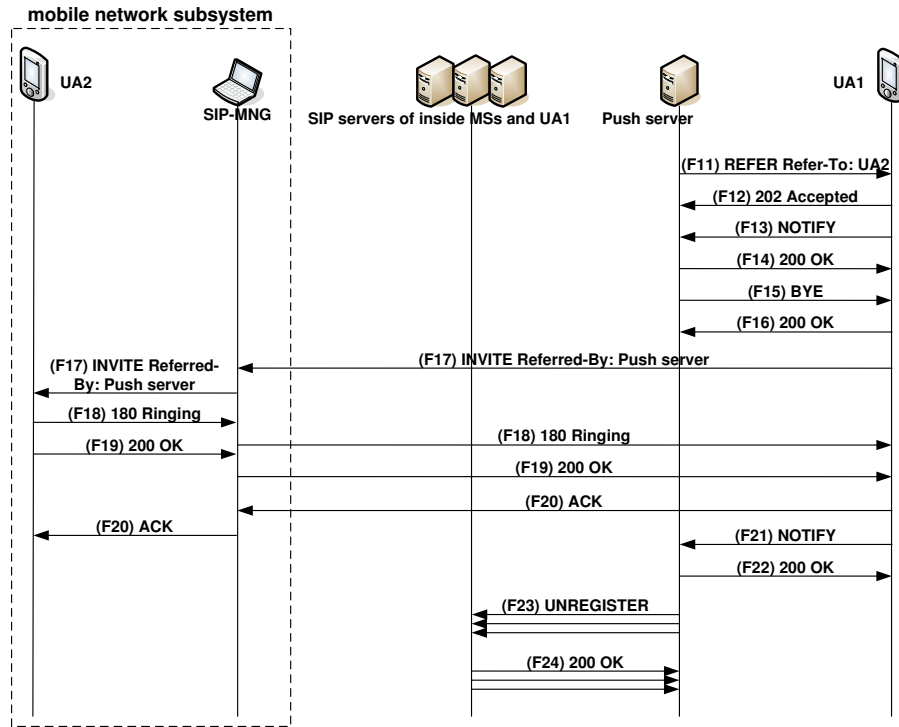


Figure 3.7: Session transfer process.

Internet. In the meantime, the push server will temporarily set up the session with UA1 to keep the session alive (F4–F7). This can prevent the SIP signaling from timeout. If this is a voice call request, to be more friendly, an option is to have the push server prepare a pre-recorded voice to tell UA1 to wait for the call to be transferred. On the other hand, when the SIP-MNG receives the short message, it will reconnect to the Internet and re-register for all MSs inside the mobile network (F8, F9) using the IP address of any active external interface of the SIP-MNG in the CONTACT field. Also, the SIP-MNG will reply an OK message to the push server via its Internet connection to update its status with the push server (F10).

- Session Transfer Process

Next, the session needs to be transferred from the push server to UA2. Fig. 3.7 depicts the message flow. The process is based on the REFER method [58] proposed by IETF to support session mobility. We comment that IETF also proposes an alternative *third-party call control (3pcc)* [54]. Requiring no special servers, both REFER and 3pcc are standard SIP solutions to support session mobility and fit our needs well. We adopt the REFER method because it requires less efforts for the push server.

After accomplishing the wake-up process, the push server will send UA1 a REFER request containing the contact information of UA2 in the Refer-To field (F11). This

message triggers UA1 to invite UA2 using the contact in the Refer-To field. UA1 then replies a SIP 202 Accepted response to the push server to indicate its approval (F12). To inform the push server that it is establishing a session with UA2, UA1 will also send a SIP NOTIFY message to the push server with an indication of “SIP/2.0 100 Trying” in the message body (F13). On receiving the NOTIFY message, the push server will respond a SIP 200 OK response (F14). The push server then sends a SIP BYE request to UA1 to terminate their session (F15). UA1 will reply a SIP 200 OK (F16). However, the dialog between the push server and UA1 will be maintained until the subscription created by the REFER is terminated.

To set up a session with UA2, UA1 sends a SIP INVITE request to UA2 containing the SDP that describes the session information of UA1. Then the rest of the SIP signalings follows the normal session setup flow (F17–F20). After the session between UA1 and UA2 is connected, UA1 will report to the push server the success of the session setup and terminate the Refer-To subscription by sending the push server a SIP NOTIFY message (F21) containing a Subscription-State header field with content of “terminated; reason=noresource”. Then, the push server will respond a SIP 200 OK message (F19). The dialog between the push server and UA1 will then be terminated. To stop acting as an agent of the mobile network subsystem, the push server may send REGISTER messages for all SIP clients inside the SIP-MNG to their SIP servers with an EXPIRE value of 0 (F23). In response, the SIP servers will sent 200 OKs (F24).

Note that a MS may simply leave the network without giving any notification. We have discussed some timeout mechanisms to determine a MS’s existence. If the SIP-MNG knows that the callee has left the network, it will not activate any external interface. Otherwise, an interface will be activated, but no MS will accept the INVITE message. So our system still works correctly.

To summarize, in our push mechanism, no change is made on the behavior of the SIP registrar and SIP proxy. SIP clients still use the standard SIP signaling. A push server can serve multiple SIP-MNGs at the same time. The REFER method is also a standard. So the system is fully compatible with existing SIP standards. Short messages and new signaling are supported by our proprietary SIP-MNG and push server. Each SIP client inside the mobile network subsystem can still use its original SIP server, SIP URI, and configuration.

3.2 *A Cross-Layer Resource Management Mechanism for Mobile Broadband Networks*

User mobility and physical rate adaptation are two important factors concerning real-time application-over-wireless network services. With user mobility, handoff between BSs/APs is inevitable. Failure to reserve sufficient bandwidths for handoff sessions may force them being dropped. Dropping on-going sessions has a very negative impact from users' perspective. While resource reservation for handoff calls is necessary, maintaining fairness among handoff calls and existing calls within a wireless network is also important. On the other hand, today's wireless transmission support multiple transmission rates to adapt to physical channel conditions. However, to keep the same level of QoS, using a lower transmission rate means that more wireless resource has to be allocated to that mobile station. Failure to reserve the required resources for roaming stations would degrade their QoS. It is a challenge to guarantee the QoS of real-time applications over a multi-rate wireless networks concerning user mobility and rate adaptation. Fortunately, today's wireless network protocols all support QoS mechanisms in the MAC layer. In this part, we show how to design a cross-layer resource management mechanism by integrating the application layer and the QoS mechanisms of wireless networks to support handoff and multi-grade QoS for real-time applications over multi-rate wireless networks. In the following, we focus on VoIP (voice over IP) and QoS mechanisms in IEEE 802.11e to design such a cross-layer mechanism. For other types of real-time applications and wireless network protocols, we believe similar technology can be applied.

We consider an IEEE 802.11e wireless network operating in the infrastructure mode to support VoIP applications. Although IEEE 802.11e supports QoS, the resource reservation and handoff handling issues for particular applications are left open to designers. In this work, we consider using SIP signaling to support VoIP over IEEE 802.11e networks.

We propose a cross-layer resource management mechanism to solve the problems caused by handoff and rate adaptation, which includes a call admission control (CAC) algorithm and a resource adjustment (RA) algorithm. The CAC algorithm is performed whenever a new call or a handoff call arrives at a QAP (Quality of Service Access Point). The CAC algorithm accepts or rejects an arriving call according to the amount of available resources versus the QoS requirements of the call. The purpose of the RA algorithm is to dynamically change bandwidth allocation among on-going calls in a QAP for better resource utilization and fairness. This is feasible because multimedia services can typically operate under different bandwidths.

In order to dynamically adjust resource allocation, we assume that VoIP calls can be supported by multiple levels of QoS. Each QoS level corresponds to a voice codec and a

PI, where PI is the period that voice data is encapsulated into packets for transmission. For most current systems, the default PI is 20 ms. Larger PIs would introduce less header overhead, but may suffer from higher delays and are more sensitive to packet loss. Given a PI and a data generation rate of λ , the amount of data to be transmitted per PI is $(\lambda \times PI + h)$, where h is the header overhead. Therefore, the bandwidth required per time unit is $(\lambda \times PI + h)/PI = \lambda + h/PI$. Clearly, a larger PI will incur less traffic.

Suppose that for each codec there are k QoS levels and each QoS level corresponds to a PI. Note when a call changes from a PI to another PI' , the difference of resource usage is $(\lambda - h/PI') - (\lambda - h/PI) = h(1/PI' - 1/PI)$. This value is only dependent of PI and PI' , but is independent of λ (and thus independent of which codec is used). Therefore, the system state of a QAP can be denoted as $\bar{s} = (s_1, s_2, s_3, \dots, s_k)$, where s_i is the number of calls served at the i th level (these calls may use different codecs). The following terminologies will be used in our CAC and RA algorithms.

- B_{total} : the total bandwidth of a QAP.
- B_{th} : the threshold of occupied bandwidth (below which new calls can be accepted).
- B_{free} : the current free bandwidth of the QAP.
- B_{deg} : the maximum available free bandwidth of the QAP after degrading all existing calls to the lowest QoS level.
- PI_{def} : the default PI for all new calls.
- W_{alloc} : the bandwidth allocated to the request call.

3.2.1 The Call Admission Control Algorithm

The CAC algorithm is to determine whether a new call or a handoff call can be accepted depending on the available bandwidth in the network. Fig. 3.8 shows the proposed CAC procedure. The caller under QAP1 first establishes a VoIP call with the callee by a SIP INVITE signal containing necessary codec and PI information. QAP1, on receiving this INVITE signal, will estimate its current available resource and possibly filter out some codecs that it cannot support due to bandwidth constraints (refer to box A in Fig. 3.8). If the callee can successfully choose a codec, QAP1 will reserve sufficient resources for the call (refer to boxes C, D and, E in Fig. 3.8). Then, the voice communication can be started. When the caller is moving away from QAP1, due to degraded signal quality, the caller will actively look for the next serving QAP by sending IEEE 802.11 *Probe Requests*. When a QAP receives a *Probe Request*, it will estimate whether it has enough available bandwidth and reply a proper IEEE 802.11 *Probe Response* (refer to box A in

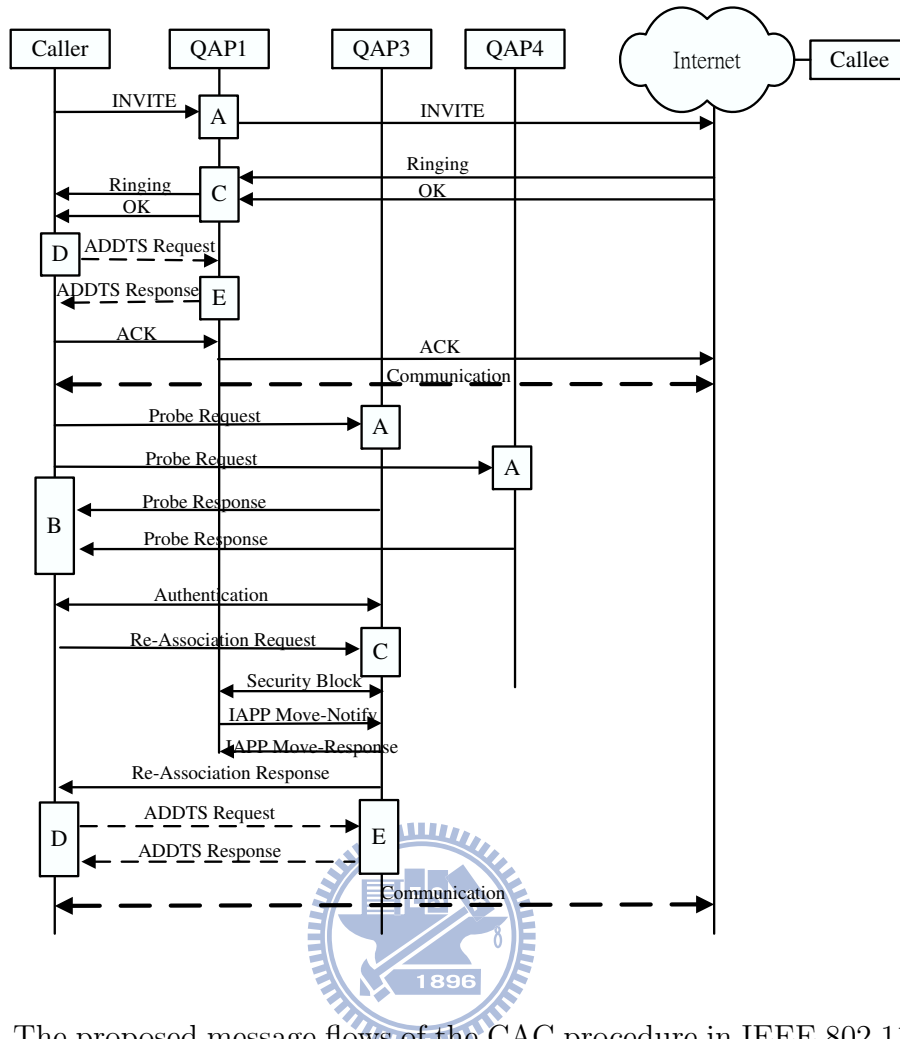


Figure 3.8: The proposed message flows of the CAC procedure in IEEE 802.11e networks.

Fig. 3.8). The caller will collect all *Probe Responses* and select a new QAP (refer to box B in Fig. 3.8). Suppose that the caller selects QAP3. It will send QAP3 an IEEE 802.11 *Re-Association Request* to trigger QAP3 to execute resource reservation (refer to box C in Fig. 3.8). In the meanwhile, QAP3 and QAP1 will exchange the caller's context by *Inter Access Point Protocol (IAPP [3])*. Via IAPP, QAP1 is informed of the departure of the caller and may execute the RA algorithm. Finally, the caller will exchange IEEE 802.11e *ADDTS Request* and *Response* with QAP3 (refer to boxes D and E in Fig. 3.8) to actually reserve the bandwidth. If these steps go through successfully, the caller and the callee can resume their voice communication. In the following, we will explain the detail actions to be taken in boxes A, B, C, D, and E.

A. Resource Estimation at the QAP

When a new call or a handoff call arrives at a QAP, the QAP will evaluate its available resource and the required resource of the call. In the new call event, the INVITE signal

Order	Information Elements
1	SSID
2	Supported Rates
8	Codec
9	Packetization Interval

(a)

Order	Information Elements
1	Timestamp
2	Beacon Interval
3	Capability Information
...	...
23	QBSS Load
24	EDCA Parameter Set
26	Codec
27	Packetization Interval

(b)

Figure 3.9: The orders of information elements in (a) Probe Request and (b) Probe Response.

from the caller will carry all codec and PI information to the callee. In the handoff call event, we propose that the caller utilizes its *Probe Requests* to convey the codec and PI information with two new IEEE 802.11 *information elements*: codec and PI. Fig. 3.9 shows the proposed orders of the information elements in *Probe Request* and *Probe Response*.

With these information, the QAP can estimate if it has sufficient resource to serve this call. To compute the required resource, we propose that each QAP keeps a *Packet Size Table (PST)* as in Table 3.2. For example, using G.726 with a sampling rate of 32 kbps and PI=20 ms, each packet is of size 154 bytes (80 bytes of voice payload, 40 bytes of IPv4/UDP/RTP/error-checking overhead, and 34 bytes of MAC/error-checking overhead). Therefore, given a call's codec, PI, and current physical rate r , we can compute the required *Medium Time (MT)* that should be reserved for the call per beacon interval (BI):

Codec	Data Rate (kbps)	Packetization Interval (ms)				
		5	10	20	30	40
G.711	64	113	154	234	314	394
G.726	16	84	94	114	134	154
	32	94	114	154	194	234
G.728	16	84	94	114	134	154
G.723.1	5.3				94	
	6.3				98	

Table 3.2: The Packet Size Table, which contains the packet sizes (in bytes) when different codecs and packetization intervals are used.

$$\begin{aligned}
MT(codec, PI, r) &= (\text{total_time_needed_per_BI}) \\
&= (\text{time_to_send_one_packet}) \times (BI/PI) \\
&\quad \times (\text{surplus_bandwidth_allowance}) \\
&= \{[(H_{RTP} + H_{UDP} + H_{IP} + H_{MAC}) + L(c, p)]/r \\
&\quad + (DIFS + \text{average}CW + PHY_header) + (SIFS + ACK)\} \\
&\quad \times (BI/PI) \times (\text{surplus_bandwidth_allowance}), \tag{3.1}
\end{aligned}$$

where $L(c, p)$ is the payload per packet when codec c and PI p are used, $\text{average}CW$ is the average contention window size seen by the QAP, and H_{RTP} , H_{UDP} , H_{IP} , and H_{MAC} are header sizes of RTP, UDP, IP, and MAC packets, respectively. The basic operation of 802.11e is shown in Fig. 3.10. Note that assuming 802.11b, the ACK and the PHY header must be transmitted at 1 Mbps. For an overview, relevant parameters of 802.11e EDCA are listed in Table 3.3. The $\text{surplus_bandwidth_allowance}$ is to take the extra medium access overhead (contention, collision, etc.) into account; in this work, we assume its value to be 1.1. This is consistent with the optimal channel utilization 0.9 found in [72] when RTS/CTS is disabled. For example, when BI is 1 sec and min_PHY_rate is 11 Mbps, if we use G.726 with rate of 32 kbps and PI of 20 msec, then $MT = [154/11(\text{bytes}/\text{Mbps}) + (50 + 70 + 192) + (10 + 248)] \times (1000/20) \times 1.1 = 37.51 \text{ ms}$.

Each QAP will keep its maximum available free bandwidth B_{deg} , which is equal to B_{free} plus the releasable resource after moving all existing calls to the lowest QoS level. That is, $B_{deg} = B_{free} + \sum_{i=1}^{k-1} h(1/PI_i - 1/PI_k)$. If a codec's required MT is larger than the QAP's B_{deg} , the QAP will drop the INVITE or the *Probe Request* silently or reply a SIP response to the caller with a status code of 480, which means "temporarily not

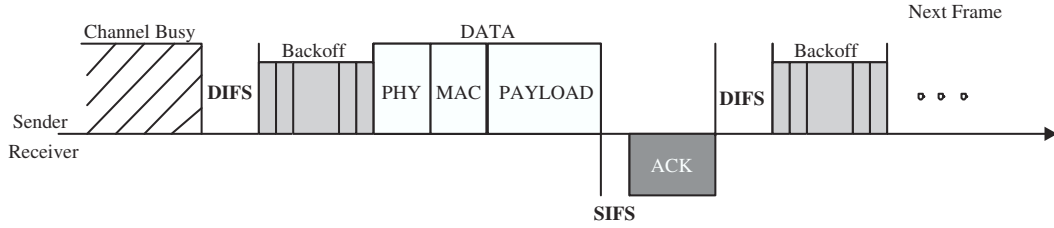


Figure 3.10: Basic operations of 802.11e EDCF.

RTP + UDP + IP header	40 bytes
MAC header for DATA	34 bytes
PHY header	192 μsec
ACK	248 μsec
DIFS	50 μsec
SIFS	10 μsec
Slot Time	20 μsec
CWmin(for voice)	7
CWmax(for voice)	15

Table 3.3: Parameters of IEEE 802.11e EDCA.

available”.

B. QAP Selection at the Caller

After scanning all channels, the caller will choose a target QAP based on various criteria, such as signal strength, codec, PI, etc. For example, we may prefer a lighter loaded QAP. Alternatively, we may choose the one with better signal quality. This is outside the scope of this work.

C. Resource Reservation at QAP

First, we will determine the codec c , PI p , and physical rate r to be used by the call. The value of r can be measured from signal quality. In the new call event, the OK signal will contain the value of c , and we will assume $p = PI_{def}$. In the handoff event, the *Re-association Request* will contain the current c and p used by the caller. Then the QAP will decide to accept or reject the call based on the following rules:

- If this is a handoff call, it will be accepted if the requested MT is no more than B_{deg} ; otherwise, the call is rejected.

degrade(c, p, r)

```
1:  $t\_PI = p$  ;
2: while (not all calls are served by  $PI_{max}$ ) do
3:   let  $X$  be the call with the smallest PI in the system;
   in case of tie, the one with the lowest physical rate is selected;
4:   change  $X$ 's PI to  $next(X.PI)$ ;
5:    $B_{free} = B_{free} + MT(X.codec, X.PI, X.rate)$ 
    $- MT(X.codec, next(X.PI), X.rate)$ ;
6:   if ( $B_{free} \geq MT(c, t\_PI, r)$ ) then
7:     return( $t\_PI$ );
8:   else if (there is no call with PI smaller than or equal to  $t\_PI$ ) then
9:      $t\_PI = next(t\_PI)$ ;
10:  end if;
11: end while;
12: return( $PI_{max}$ );
```

Figure 3.11: The bandwidth degrade algorithm.

- If this is a new call, there are two cases:
 - If $MT(c, PI_{max}, r) \leq B_{deg}$ and $B_{deg} > (B_{total} - B_{th})$, the call is accepted directly.
 - If $MT(c, PI_{max}, r) \leq B_{deg}$ but $B_{deg} \leq (B_{total} - B_{th})$, the call is accepted with a probability P_r .

Note that the selection of P_r can be based on the *DCRS* (*Dynamic Channel Reservation Scheme*) proposed in [31]. In this work, we will only consider adjusting PI for handoff calls, although adjusting codec is also possible.

If the call cannot be accepted, the QAP will drop the OK silently (for new call) or reply the *Re-Association Response* to the caller with a status code of 37, which means “The request has been declined.” (for handoff call). If the call can be accepted, we will check if $MT(c, p, r) \leq B_{free}$. If so, the selected codec and PI will be relayed to the caller via an OK (for new call) or a *Re-association Response* (for handoff call). Otherwise, the current available resource is not able to support the request and we will call function *degrade*(c, p, r) in Fig. 3.11. The function will repeatedly select an existing call to reduce its QoS level. The call with the best QoS level will be degraded first. If there are multiple candidates, the one with the lowest physical rate will be degraded first. Function *next*() will return the next QoS level. This is repeated until sufficient resources are released.

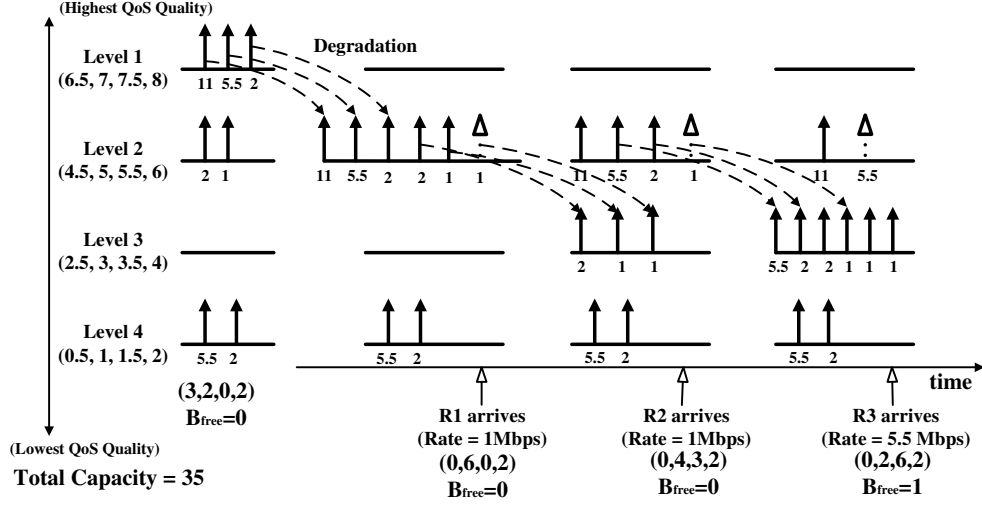


Figure 3.12: An example of bandwidth degrade.

Fig. 3.12 shows an example. Suppose that there are $k = 4$ QoS levels, and the current system state is $(3,2,0,2)$. Also suppose that the required resources for these QoS levels are $(8, 7.5, 7, 6.5)$, $(6, 5.5, 5, 4.5)$, $(4, 3.5, 3, 2.5)$, and $(2, 1.5, 1, 0.5)$, respectively (the four numbers map to four physical rates in an ascending order). The total capacity is equal to 35. So, there is no resource remaining. Suppose that an incoming call requests a QoS level of 2 (physical rate = 1 Mbps). As the resource required is 6, we need to degrade three calls from QoS level 1 to level 2. The next incoming call also requests a QoS level of 2 (physical rate = 1 Mbps). The resource required is 6, too. We need to degrade three of level-2 calls. The calls with lower transmission rates should be degraded first, so we move two calls with 1 Mbps and one with 2 Mbps to level 3. Then the system state will change to $(0, 4, 3, 2)$. The last incoming call requests a QoS level of 2 (physical rate = 5.5 Mbps). According to the algorithm, we move three level-2 calls to level 3. The final system state is $(0, 2, 6, 2)$.

D. ADDTS Request by the Caller

After determining the codec and PI, the caller will send a bidirectional *ADDTS Request* to the QAP by including a TSPEC element to request for resources. We suggest to convey VoIP service requirements by the following fields in TSPEC:

- `Minimum_Data_Rate` = the acceptable longest PI of the corresponding codec.
- `Mean_Data_Rate` = the PI selected by the callee.
- `Maximum_Data_Rate` = the acceptable shortest PI.
- `Medium_Time` = the codec selected by the callee.

E. ADDTS Response by the QAP

According to the caller's *ADDTS Request* and the Packet Size Table, QAP can compute the required MT following Eq.(3.1). Each QAP keeps the following variables:

- $TXOPBudget[AC_i]$ = The remaining bandwidth that can be allocated to AC_i , $i = 0..3$.
- $TxAddDn[AC_i][TSID]$ = The admitted MT for stream TSID of AC_i in the downlink direction.
- $TxAdUp[AC_i][TSID]$ = The admitted MT for stream TSID of AC_i in the uplink direction.
- $TxAddDn[AC_i]$ = This value is set to $\sum_{\forall TSID} TxAddDn[AC_i][TSID]$, to record the overall resource allocated to AC_i in the downlink direction.
- $TxUsedDn[AC_i]$ = The summation of used MT of all downlink streams of AC_i .

Initially, $TXOPBudget[AC_i]$ contains all the bandwidth (in terms of MT) that is reserved for AC_i . Whenever a new stream is added, the corresponding resource is subtract from $TXOPBudget[AC_i]$, and the resource is assigned to $TxAddDn[AC_i][TSID]$ and/or $TxAdUp[AC_i][TSID]$. Also, each QSTA (Quality of Service Station) should keep the following variables:

- $TxAdUp[AC_i][TSID]$ = The admitted MT for stream TSID of AC_i in the uplink direction in this QSTA per BI.
- $TxAdUp[AC_i]$ = This value is set to $\sum_{\forall TSID} TxAdUp[AC_i][TSID]$, to record the overall resource allocated to AC_i in the uplink direction.
- $TxUsedUp[AC_i]$ = The summation of used MT of all uplink streams of AC_i .

Resource reservation at QAP is done as follows. First, we compute the value of $TXOPBudget[AC_i] - 2 \times MT(c, p, r)$. If the value is non-negative, there is sufficient resource to support this call and we can set variables as follows:

$$TXOPBudget[AC_i] = TXOPBudget[AC_i] - 2 \times MT(c, p, r);$$

$$TxAddDn[AC_i][TSID] = MT(codec, PI, PHY);$$

$$TxAdUp[AC_i][TSID] = MT(codec, PI, PHY);$$

$$TxAddDn[AC_i] = TxAddDn[AC_i] + TxAddDn[AC_i][TSID];$$

Resource_Adjustment()

- 1: On a call X moving to a lower rate r :
 $B_{free} = B_{free} + MT(X.codec, X.PI, X.rate);$
if($B_{free} < MT(X.codec, X.PI, r)$)
 $degrade(X.codec, X.PI, r);$
 - 2: On a call X leaving:
 $B_{free} = B_{free} + MT(X.codec, X.PI, X.rate);$
 $upgrade();$
 - 3: On a call X moving to a higher rate r :
 $B_{free} = B_{free} + MT(X.codec, X.PI, X.rate) - MT(X.codec, X.PI, r);$
 $upgrade();$
-

Figure 3.13: The RA algorithm.

Up to this point, the admitted resources have been guaranteed. The QAP will reply an *ADDTS Response* to the caller with the Mean_Data_Rate=PI and Medium_Time=MT(c,p,r) in TSPEC. If there is no sufficient resource, then an *ADDTS Response* is replied with Medium_Time=0.

At the caller's side, if an *ADDTS response* with a positive Medium_Time is received, the QSTA will set its $TxAUp[AC_i][TSID] = \text{Medium_Time}$, retrieves the PI in the Mean_Data_Rate field, and passes it to the upper layer VoIP application program. Otherwise, the call is considered rejected. In both cases, the caller should reply a response signal with the proper status code to the callee.

3.2.2 The Resource Adjustment Algorithm

Fairness among existing users and handoff users is an important issue. The goal of resource adjustment is to re-allocate bandwidth to calls for fairness. The RA algorithm may be triggered by the following two events: departure of calls and transmission rate change of existing calls (refer to Fig. 3.13). On events that a call moves to a lower rate, the function $degrade(c,p,r)$ will be called if there is no sufficient resource. On events that a call departs or moves to a higher rate, the value of B_{free} will be updated, and then the function $upgrade()$ in Fig. 3.14 will be invoked. This function will repeatedly select an existing call to upgrade its QoS level. The call with the worst QoS level will be upgraded first. If there are multiple candidates, the one with the highest physical rate will be upgraded first. Function $prev()$ will return the previous QoS level. This is repeated until B_{free} is not enough to upgrade any existing call.

upgrade()

```
1: while (TRUE) do
2:   let  $X$  be the call with the largest PI in the system;
   in case of tie, the one with the highest physical rate is selected;
3:   if  $B_{free} \geq MT(X.codec, prev(X.PI), X.rate) - MT(M.codec, X.PI, X.rate)$  then
4:     change  $X$ 's PI to  $prev(X.PI)$ ;
5:      $B_{free} = B_{free} - MT(X.codec, prev(X.PI), X.rate) + MT(M.codec, X.PI, X.rate)$ ;
6:   else
7:     return;
8:   end if;
9: end while;
```

Figure 3.14: The bandwidth upgrade algorithm.

Fig. 3.15 shows an example. Suppose that there are $k = 4$ QoS levels, and the current system state is $(4, 1, 1, 3)$. The resource requirement is the same as the example in Fig. 3.12. Let the total capacity be 41. Suppose that a level 1 call leaves the network (physical rate = 1 Mbps), releasing a bandwidth of 2. The released bandwidth can upgrade the call at QoS level 4 with rate 11 Mbps to level 3. The system state after upgrade is $(4, 1, 2, 1)$. Next, a level-2 call with rate 2Mbps leaves, releasing a bandwidth of 5.5. This can upgrade the only call at level 4 to level 3, resulting a system state of $(4, 1, 2, 0)$.

3.2.3 Analysis

In this section, we derive an analytical model to evaluate the performance of our QoS mechanisms. Our goal is to analyze the blocking probability of new calls, the dropping probability of handoff calls, and the call dropping probability due to change of transmission rates. Without loss of generality, we assume all calls use the same G.726 codec at the default rate 32 kbps. Thus, during a degrade or upgrade process, calls will only change their PIs, but not codec. Suppose that there are m PIs, PI_1, PI_2, \dots, PI_m (in an ascending order), and y transmission rates, R_1, R_2, \dots, R_y (in a descending order).

Due to mobility, the rate change of a QSTA is modeled by the state diagram in Fig. 3.16. From each state, a QSTA can transit to a higher or a lower rate with a rate ν following a Poisson distribution. In each QAP, new and handoff calls arrive by Poisson distributions with rates $y \cdot \lambda_n$ and $y \cdot \lambda_h$, respectively. These rates are evenly distributed to calls of all physical rates R_1, R_2, \dots, R_y . Call holding time and cell residence time are exponentially distributed with means of $1/\mu_h$ and $1/\mu_r$, respectively. Thus, the channel occupancy time of a call is exponentially distributed with mean $1/\mu = 1/(\mu_h + \mu_r)$. The

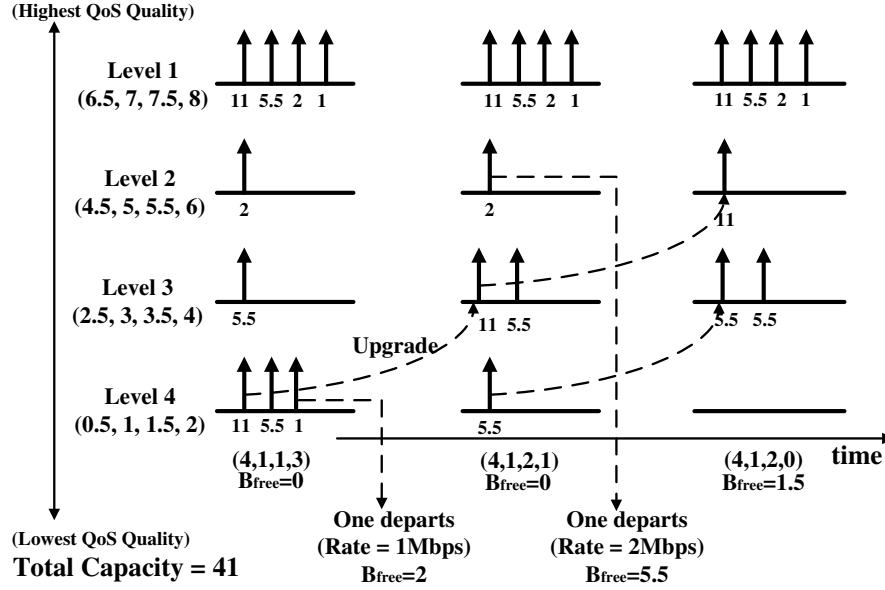


Figure 3.15: An example of bandwidth upgrade.

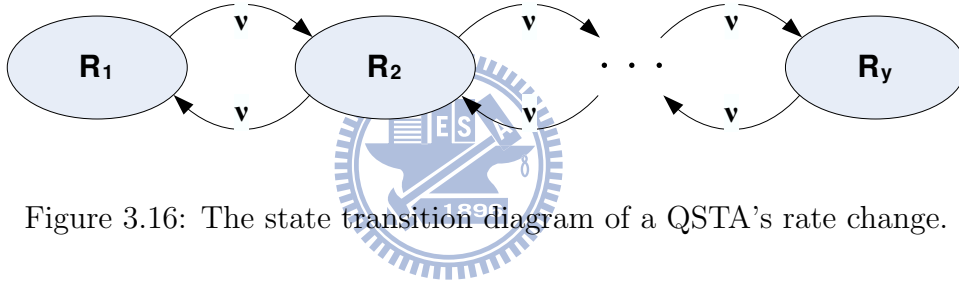


Figure 3.16: The state transition diagram of a QSTA's rate change.

required bandwidth of a call with PI_{max} at the transmission rate R_i is denoted by Φ_i .

According to our CAC algorithm, computation of accepting or rejecting a call is all based on the assumption that all calls can be degraded to the lowest QoS level. In other words, a QAP drops or blocks a call when all existing calls use PI_{max} and the sum of their used bandwidth meets some conditions. Therefore, to obtain blocking and dropping probabilities, we can assume that all calls use PI_{max} .

For simplicity, we assume that a QAP can support $y = 4$ physical rates, 11, 5.5, 2 and 1 Mbps. For a bi-direction voice stream, assuming $BI = 1sec$, $surplus = 1.1$, and $PI_{max} = 40ms$, the required MT per BI of a call under each rate is listed in Table 3.4.

Our system can be modeled by a $y = 4$ dimensional Markov process. Each system

Transmission rate (Mbps)	11	5.5	2	1
Occupied MT (sec)/per session	0.041	0.050	0.083	0.134

Table 3.4: The required MT of a bi-directional voice call under different physical rates under our analytical model.

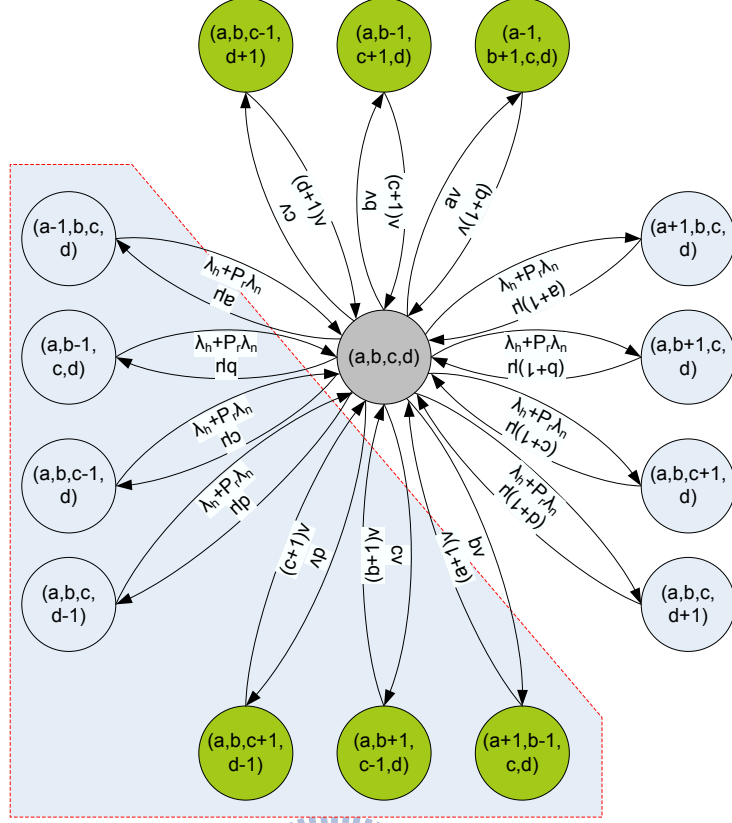


Figure 3.17: Generic state transitions under our analytical model.

state is written as (n_1, n_2, n_3, n_4) , where n_i is the number of calls at rate R_i , $i = 1 \dots 4$. For each state (n_1, n_2, n_3, n_4) , there 14 possible state transitions, as shown as Fig. 3.17, where $n_1 = a$, $n_2 = b$, $n_3 = c$, $n_4 = d$. Horizontal transitions are caused by call arrival or departure events. The arrival rates are all modeled by $\lambda_h + P_r \lambda_n$. The departure rate for rate R_i is $n_i \mu$. For ease of presentation, we let $P_r = 1$ when $n_1 \Phi_1 + n_2 \Phi_2 + n_3 \Phi_3 + n_4 \Phi_4 < B_{th}$; otherwise, new calls are accepted with a probability P_r as defined in Sec. 3.1. Vertical transitions are caused by transmission rate change.

A simplified two-dimension Markov process is shown in Fig. 3.18 for the case of $y = 2$. The states marked by gray are those with $P_r = 1$, where all new calls can be accepted. Under other states, a new call will be dropped with a fixed probability $(1 - P_r)$.

Based on above state transition diagram, we can derive the steady-state probability P_{n_1, n_2, \dots, n_y} of each state. There are four cases:

Case I: For the state such that $n_1 = n_2 = \dots = n_y = 0$,

$$y(\lambda_h + \lambda_n) P_{n_1, n_2, \dots, n_y} = \mu \sum_{i=1}^y P_{n_1, n_2, \dots, n_i+1, \dots, n_y}. \quad (3.2)$$

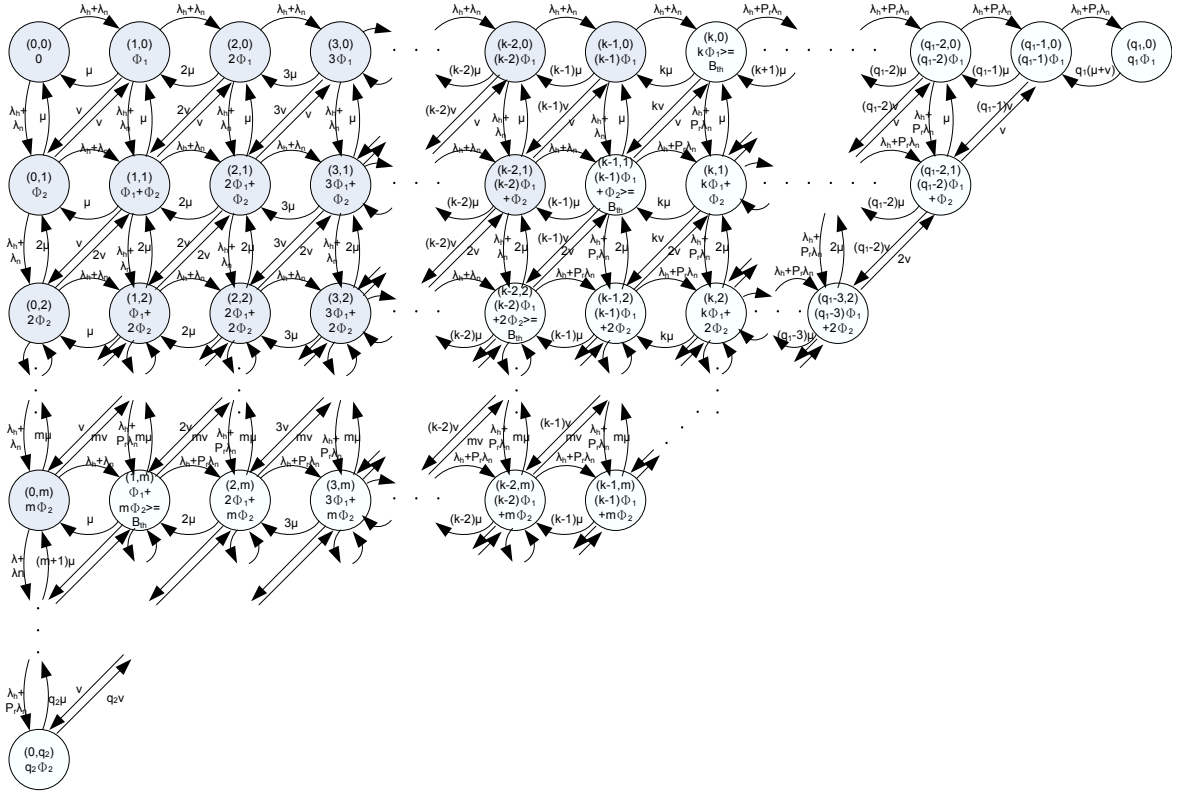


Figure 3.18: A state transition example with $y = 2$ (q_1 and q_2 are the maximum numbers of calls that can be accommodated with PI_{max} at rates R_1 and R_2 , respectively).

Case II: For states such that $\sum_{i=1}^y (n_i \Phi_i) < B_{th}$,

$$\begin{aligned}
 & \left[y(\lambda_h + \lambda_n) + \left(\sum_{i=1}^y n_i \right) \mu + \left(\sum_{i=1}^{y-1} n_i + \sum_{i=2}^y n_i \right) \nu \right] P_{n_1, n_2, \dots, n_y} \\
 = & \sum_{i=1}^y \left[(n_i + 1) \mu P_{n_1, n_2, \dots, n_i+1, \dots, n_y} + (\lambda_h + \lambda_n) P_{n_1, n_2, \dots, n_i-1, \dots, n_y} \right] \\
 & + \sum_{i=2}^y \left[(n_i + 1) \nu P_{n_1, n_2, \dots, n_{i-1}-1, n_i+1, \dots, n_y} \right] \\
 & + \sum_{i=1}^{y-1} \left[(n_i + 1) \nu P_{n_1, n_2, \dots, n_i+1, n_{i+1}-1, \dots, n_y} \right]. \tag{3.3}
 \end{aligned}$$

Case III: For states such that $\sum_{i=1}^y (n_i \Phi_i) \geq B_{th}$ and $\sum_{i=1}^y (n_i \Phi_i) + \Phi_y < B_{total}$,

$$\begin{aligned}
& \left[y(\lambda_h + P_r \lambda_n) + \left(\sum_{i=1}^y n_i \right) \mu + \left(\sum_{i=1}^{y-1} n_i + \sum_{i=2}^y n_i \right) \nu \right] P_{n_1, n_2, \dots, n_y} \\
= & \sum_{i=1}^y [(n_i + 1) \mu P_{n_1, n_2, \dots, n_i+1, \dots, n_y}] \\
& + \sum_{i=1}^y [(\lambda_h + (I_i + \bar{I}_i P_r) \lambda_n) P_{n_1, n_2, \dots, n_i-1, \dots, n_y}] \\
& + \sum_{i=2}^y [(n_i + 1) \nu P_{n_1, n_2, \dots, n_i-1-1, n_i+1, \dots, n_y}] \\
& + \sum_{i=1}^{y-1} [(n_i + 1) \nu P_{n_1, n_2, \dots, n_i+1, n_{i+1}-1, \dots, n_y}], \tag{3.4}
\end{aligned}$$

where for $z \in \{1, \dots, y\}$

$$I_z = \begin{cases} 1, & \sum_{i=1}^y (n_i \Phi_i) - \Phi_z < B_{th} \\ 0, & \text{otherwise} \end{cases}.$$

Case IV: For states such that $\sum_{i=1}^y (n_i \Phi_i) \leq B_{total}$ and $\sum_{i=1}^y (n_i \Phi_i) - \Phi_y \geq B_{th}$,

$$\begin{aligned}
& \left[\left(\sum_{i=1}^y I_i \right) (\lambda_h + P_r \lambda_n) + \left(\sum_{i=1}^y n_i \right) \mu + \left(\sum_{i=1}^{y-1} n_i + \sum_{i=2}^y n_i \right) \nu \right] P_{n_1, n_2, \dots, n_y} \\
= & \sum_{i=1}^{y-1} [I_i (n_i + 1) (\mu + \bar{I}_{i+1} \nu) P_{n_1, n_2, \dots, n_i+1, \dots, n_y}] \\
& + I_y (n_y + 1) \mu P_{n_1, n_2, \dots, n_{y-1}, n_y+1} \\
& + (\lambda_h + P_r \lambda_n) \sum_{i=1}^y P_{n_1, n_2, \dots, n_i-1, \dots, n_y} \\
& + \sum_{i=2}^y [I_{i-1, i} (n_i + 1) \nu P_{n_1, n_2, \dots, n_i-1-1, n_i+1, \dots, n_y}] \\
& + \sum_{i=1}^{y-1} [(n_i + 1) \nu P_{n_1, n_2, \dots, n_i+1, n_{i+1}-1, \dots, n_y}], \tag{3.5}
\end{aligned}$$

where for $z \in \{1, \dots, y\}$

$$I_z = \begin{cases} 1, & \sum_{i=1}^y (n_i \Phi_i) + \Phi_z \leq B_{total} \\ 0, & \text{otherwise} \end{cases}$$

and for $(m, n) \in \{(1, 2), (2, 3), \dots, (y-1, y)\}$

$$I_{m, n} = \begin{cases} 1, & \sum_{i=1}^y (n_i \Phi_i) - \Phi_m + \Phi_n \leq B_{total} \\ 0, & \text{otherwise} \end{cases}.$$

Let P_b be the blocking probability of new calls, P_d be the dropping probability of handoff calls, and P_{td} be the call dropping probability due to change of transmission rates. Given any system state $\bar{n} = (n_1, n_2, \dots, n_y)$, let the bandwidth requirement $\tau(\bar{n}) = \sum_{i=1}^y (n_i \Phi_i)$. We can derive:

$$P_b = \sum_{\tau(\bar{n}) \geq B_{th}} P_{n_1, n_2, \dots, n_y} \quad (3.6)$$

$$P_d = \frac{1}{y} \sum_{i=1}^y \left(\sum_{\tau(\bar{n}) + \Phi_i > B_{total}} P_{n_1, n_2, \dots, n_y} \right) \quad (3.7)$$

$$P_{td} = \sum_{i=1}^{y-1} \left[\sum_{\tau(\bar{n}) - \Phi_i + \Phi_{i+1} > B_{total}} \left(\frac{n_i}{\sum_{i=1}^{y-1} n_i + \sum_{i=2}^y n_i} P_{n_1, n_2, \dots, n_y} \right) \right]. \quad (3.8)$$

To compute P_b , P_d , and P_{td} , we have to solve the steady-state probabilities P_{n_1, n_2, \dots, n_y} . This can be done by the recursive technique proposed by Herzog *et al.* [21], which states that there exists a subset of the state probabilities, called *boundaries*, such that all other states can be expressed as linear combinations of the boundary states. Therefore, we can determine the boundaries first and then derive the expressions for all remaining state probabilities as functions of the boundary values. This can significantly reduce the complexity of solving of P_{n_1, n_2, \dots, n_y} as compared to traditional matrix inversion techniques. It has been shown to be suitable to solve a wide class of queuing problems.

First, we choose all states (n_1, n_2, \dots, n_y) such that $n_y = 0$ as boundaries. According to [21], we can rewrite the state probabilities as:

$$\begin{aligned} & P_{n_1, n_2, \dots, n_y} \\ = & \sum_{\alpha_{y-1}=0}^{\frac{B_{total}}{\Phi_{y-1}}} \dots \sum_{\alpha_2=0}^{\lfloor \frac{B_{total}}{\Phi_2} \rfloor} \sum_{\alpha_1=0}^{\lfloor \frac{B_{total}}{\Phi_1} \rfloor} C_{n_1, n_2, \dots, n_y}^{\alpha_1, \alpha_2, \dots, \alpha_{y-1}} P_{\alpha_1, \alpha_2, \dots, \alpha_{y-1}, \alpha_y=0} \end{aligned} \quad (3.9)$$

$$C_{n_1, n_2, \dots, n_{y-1}, n_y=0}^{\alpha_1, \alpha_2, \dots, \alpha_{y-1}} = \begin{cases} 1, & \text{if } n_1 = \alpha_1, n_2 = \alpha_2, \dots, \text{ and } n_{y-1} = \alpha_{y-1} \\ 0, & \text{otherwise} \end{cases}.$$

The coefficients $C_{n_1, n_2, \dots, n_y}^{\alpha_1, \alpha_2, \dots, \alpha_{y-1}}$ for $n_y \neq 0$ can be solved recursively. Next, we rewrite Eq.s (3.2)–(3.5) by:

$$\begin{aligned}
B_{n_1, n_2, \dots, n_y} P_{n_1, n_2, \dots, n_y} &= \sum_{i=1}^y A_{n_1, n_2, \dots, n_y}^{r_i} P_{n_1, n_2, \dots, n_i+1, \dots, n_y} \\
&+ \sum_{i=1}^y A_{n_1, n_2, \dots, n_y}^{l_i} P_{n_1, n_2, \dots, n_i-1, \dots, n_y} \\
&+ \sum_{i=2}^y A_{n_1, n_2, \dots, n_y}^{u_i} P_{n_1, n_2, \dots, n_{i-1}-1, n_i+1, \dots, n_y} \\
&+ \sum_{i=1}^{y-1} A_{n_1, n_2, \dots, n_y}^{d_i} P_{n_1, n_2, \dots, n_i+1, n_{i+1}-1, \dots, n_y}, \tag{3.10}
\end{aligned}$$

where the coefficients $A_{n_1, n_2, \dots, n_y}^{r_i}$, $A_{n_1, n_2, \dots, n_y}^{l_i}$, $A_{n_1, n_2, \dots, n_y}^{u_i}$, $A_{n_1, n_2, \dots, n_y}^{d_i}$, and B_{n_1, n_2, \dots, n_y} are abbreviations of those in Eq.s (3.2)–(3.5). Then we rewrite Eq. (3.10) as:

$$\begin{aligned}
P_{n_1, n_2, \dots, n_{y-1}, n_y+1} &= \frac{B_{n_1, n_2, \dots, n_y} P_{n_1, n_2, \dots, n_y} - \sum_{i=1}^{y-1} A_{n_1, n_2, \dots, n_y}^{r_i} P_{n_1, n_2, \dots, n_i+1, \dots, n_y}}{A_{n_1, n_2, \dots, n_y}^{r_y}} \\
&- \frac{\sum_{i=1}^y A_{n_1, n_2, \dots, n_y}^{l_i} P_{n_1, n_2, \dots, n_i-1, \dots, n_y}}{A_{n_1, n_2, \dots, n_y}^{r_y}} \\
&- \frac{\sum_{i=2}^y A_{n_1, n_2, \dots, n_y}^{u_i} P_{n_1, n_2, \dots, n_{i-1}-1, n_i+1, \dots, n_y}}{A_{n_1, n_2, \dots, n_y}^{r_y}} \\
&- \frac{\sum_{i=1}^{y-1} A_{n_1, n_2, \dots, n_y}^{d_i} P_{n_1, n_2, \dots, n_i+1, n_{i+1}-1, \dots, n_y}}{A_{n_1, n_2, \dots, n_y}^{r_y}}. \tag{3.11}
\end{aligned}$$

After some manipulation, Eq. (3.11) can be further rewritten as:

$$\begin{aligned}
P_{n_1, n_2, \dots, n_{y-1}, n_y} &= \frac{B_{n_1, n_2, \dots, n_{y-1}} P_{n_1, n_2, \dots, n_{y-1}} - \sum_{i=1}^{y-1} A_{n_1, n_2, \dots, n_{y-1}}^{r_i} P_{n_1, n_2, \dots, n_i+1, \dots, n_{y-1}}}{A_{n_1, n_2, \dots, n_{y-1}}^{r_y}} \\
&- \frac{\sum_{i=1}^y A_{n_1, n_2, \dots, n_{y-1}}^{l_i} P_{n_1, n_2, \dots, n_i-1, \dots, n_{y-1}}}{A_{n_1, n_2, \dots, n_{y-1}}^{r_y}} \\
&- \frac{\sum_{i=2}^y A_{n_1, n_2, \dots, n_{y-1}}^{u_i} P_{n_1, n_2, \dots, n_{i-1}-1, n_i+1, \dots, n_{y-1}}}{A_{n_1, n_2, \dots, n_{y-1}}^{r_y}} \\
&- \frac{\sum_{i=1}^{y-1} A_{n_1, n_2, \dots, n_{y-1}}^{d_i} P_{n_1, n_2, \dots, n_i+1, n_{i+1}-1, \dots, n_{y-1}}}{A_{n_1, n_2, \dots, n_{y-1}}^{r_y}}. \tag{3.12}
\end{aligned}$$

For each fixed state $(\alpha_1, \alpha_2, \dots, \alpha_{y-1}, 0)$, if we let $P_{\bar{n}} = 1$ and $P_{\bar{n}'} = 0$ for all $\bar{n}' \neq \bar{n}$, from Eq.s (3.9) and (3.12), we can obtain:

$$\begin{aligned}
C_{n_1, n_2, \dots, n_y}^{\alpha_1, \alpha_2, \dots, \alpha_{y-1}} &= \frac{B_{n_1, n_2, \dots, n_{y-1}} C_{n_1, n_2, \dots, n_{y-1}}^{\alpha_1, \alpha_2, \dots, \alpha_{y-1}} - \sum_{i=1}^{y-1} A_{n_1, n_2, \dots, n_{y-1}}^{r_i} C_{n_1, n_2, \dots, n_i+1, \dots, n_{y-1}}^{\alpha_1, \alpha_2, \dots, \alpha_{y-1}}}{A_{n_1, n_2, \dots, n_{y-1}}^{r_y}} \\
&= \frac{\sum_{i=1}^y A_{n_1, n_2, \dots, n_{y-1}}^{l_i} C_{n_1, n_2, \dots, n_i-1, \dots, n_{y-1}}^{\alpha_1, \alpha_2, \dots, \alpha_{y-1}}}{A_{n_1, n_2, \dots, n_{y-1}}^{r_y}} \\
&= \frac{\sum_{i=2}^y A_{n_1, n_2, \dots, n_{y-1}}^{u_i} C_{n_1, n_2, \dots, n_{i-1}-1, n_i+1, \dots, n_{y-1}}^{\alpha_1, \alpha_2, \dots, \alpha_{y-1}}}{A_{n_1, n_2, \dots, n_{y-1}}^{r_y}} \\
&= \frac{\sum_{i=1}^{y-1} A_{n_1, n_2, \dots, n_{y-1}}^{d_i} C_{n_1, n_2, \dots, n_i+1, n_{i+1}-1, \dots, n_{y-1}}^{\alpha_1, \alpha_2, \dots, \alpha_{y-1}}}{A_{n_1, n_2, \dots, n_{y-1}}^{r_y}} \tag{3.13}
\end{aligned}$$

for all combinations of n_1, n_2, \dots, n_y .

After obtaining all coefficients $C_{n_1, n_2, \dots, n_y}^{\alpha_1, \alpha_2, \dots, \alpha_{y-1}}$, the probabilities of boundaries can be derived by solving the remaining unused $\lfloor \frac{B_{total}}{\Phi_1} \rfloor \times \lfloor \frac{B_{total}}{\Phi_2} \rfloor \times \dots \times \lfloor \frac{B_{total}}{\Phi_{y-1}} \rfloor$ independent equations in Eq. (3.10) as well as the normalizing condition:

$$\sum_{n_1} \sum_{n_2} \dots \sum_{n_y} P_{n_1, n_2, \dots, n_y} = 1. \tag{3.14}$$

Having solved the boundaries, all steady-state probabilities P_{n_1, n_2, \dots, n_y} can be determined from (3.9). Thus, P_b , P_d , and P_{td} can then be derived.

3.3 Mobility Management for Mobile Broadband Networks

Supporting user and device mobilities is a critical issue for wireless networks since continuous network connectivity is highly desirable for most services. However, supporting voice and multimedia with mobility implies that the total handoff latency must be small to guarantee their QoS. The latency for VoIP should not exceed 50 ms to prevent excessive jitter [65], while streaming video/audio applications cannot tolerate a latency more than 150 ms [71]. So, in this part, to guarantee the QoS of services, we propose a novel seamless post-handoff mechanism for IEEE 802.11 wireless networks which support IEEE 802.11i security standard. Our approach consists of a *Dynamic Tunnel Establishing* procedure and a *seamless handoff* mechanism. Both intra- and inter-subnet handoff cases are considered in our approach. Similar concept can be applied in other types of wireless networks with security support to provide wireless users continuous connectivity.

In the existing 802.1x authentication, users are not allowed to transmit/receive normal data via the new AP before the authentication succeeds. Considering that blocking normal data access can be done later on, we propose to allow a roaming MS to execute the 802.1x authentication and normal data access simultaneously for a short period of time. However,

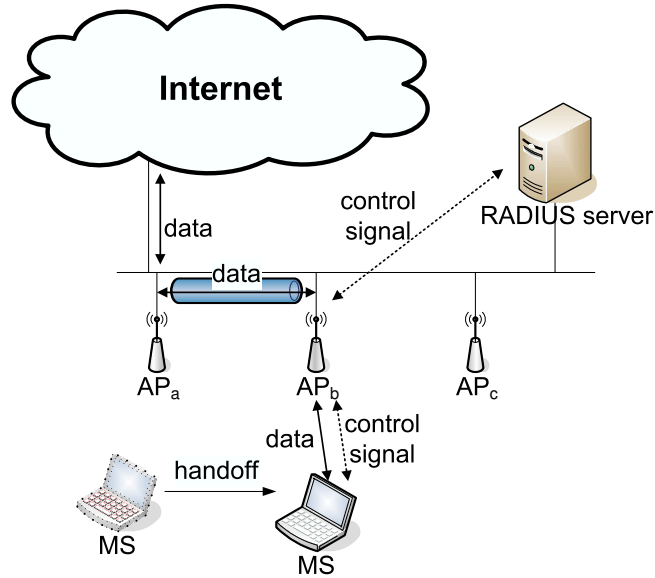


Figure 3.19: System architecture.

this may cause a security loophole. Hence, we enforce the MS to access the Internet via its previous AP before handoff completes. Since the previous AP has authenticated the MS, it can check if the MS is a legal user and provide the MS secure wireless access by using its prior data encryption key. Our approach consists of a *Dynamic Tunnel Establishing* procedure and a *seamless handoff* method. Dynamic Tunnel Establishing is for each AP to construct trusted tunnels with its neighbor APs. In the seamless handoff method, we propose to allow the roaming MS to access the Internet via its previous AP by the tunnel between the new AP and the previous one during the Execution, DHCP, and Upper Layer Adjustment phases of a handoff. Fig. 3.19 shows our proposed seamless handoff architecture, where an MS is moving from AP_a to AP_b . When the MS is authenticating with the RADIUS server via the new AP, AP_b , it also continues to access the Internet via its previous AP, AP_a . Therefore, we can achieve a seamless handoff for roaming MSs. Below, we will give an overview of our Dynamic Tunnel Establishing procedure and seamless handoff method.

Fig. 3.20 shows the Dynamic Tunnel Establishing procedure, which is triggered by the receipt of IEEE 802.11 *reassociation request* or IAPP *Move-Notify* message of AP_b . On receiving one of these two messages, AP_b finds that AP_a is its neighbor and then tries to establish a tunnel with AP_a . Before sending the request to AP_a , AP_b first verifies if AP_a is a trusted AP by querying its RADIUS server. If the verification succeeds, AP_b then sends AP_a a *tunlestab-request*. This *tunlestab-request* message asks AP_a to construct a layer-two tunnel if both APs are in the same subnet; otherwise, it asks AP_a to construct a layer-three tunnel. ARP (Address Resolution Protocol) can be used to determine whether both APs are in the same subnet or not. On the receipt of the *tunlestab-request* message,

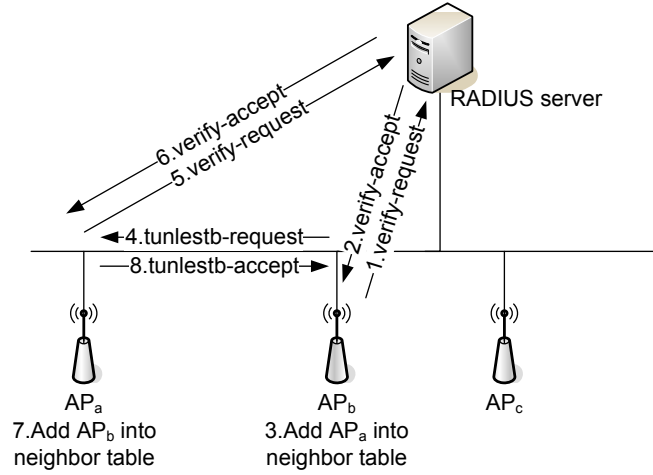


Figure 3.20: Proposed Dynamic Tunnel Establishing procedure.

AP_a will also verify if AP_b is a trusted AP. If yes, AP_a will agree to establish the tunnel with AP_b .

Since the Dynamic Tunnel Establishing procedure is executed after the very first roaming MS handoffs from one AP to another, the first MS can not benefit from the seamless handoff method. However, later roaming MSs can all enjoy such tunneling services.

Next, let's see how the seamless handoff method works. Fig. 3.21 shows our seamless handoff mechanism, in which MS_i has a data connection with a corresponding node (CN) and it is moving from its old AP to a new AP. Here we assume that the SIP (Session Initiation Protocol) mobility [66] is used in the Upper Layer Adjustment phase. In our method, we do not propose an enhancement to the Probe-and-Decision phase because existing schemes already do well (such as the NG-based selective scanning [57, 23] and the SyncScan [46]). We can adopt one of them in the Probe-and-Decision phase (H1). After deciding the new AP, MS_i will send it a reassociation request message (H2). Upon the receipt of the reassociation request from MS_i , the new AP will be ready to relay data between the old AP and MS_i using the existing tunnel between it and the old AP. Here both the new AP and the old AP will set a timer $T1$ as the threshold time to provide MS_i Internet access. Then, the new AP will reply MS_i a reassociation response and the 802.1x authentication starts (H6). Once $T1$ times out, if the 802.1x authentication does not complete, the relay will be prohibited by the new AP and the old AP. Notice that, before MS_i and the new AP derives a new data encryption key, MS_i will use the old key to encrypt data sent to the old AP. The new AP only tunnels the data to the old AP or forwards the encrypted data to MS_i . Once the 802.1x authentication and the four-way handshake complete, the new AP will close $T1$ (H7) and inform the old AP that the authentication has succeeded (H8). If this is an intra-subnet handoff, the

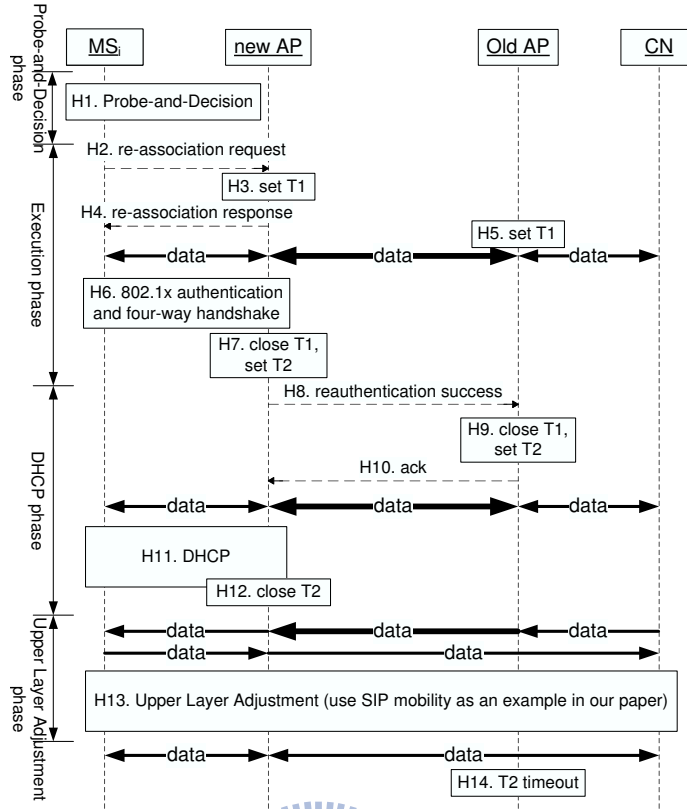


Figure 3.21: Proposed seamless handoff mechanism.

seamless handoff procedure finishes; otherwise, this is an inter-subnet handoff and the new AP and the old AP will set a timer $T2$ (in H7 and H9, respectively) to continue data relay for MS_i to maintain continuous network connectivity during executing remaining handoff procedures. The tunnel type between the old AP and the new AP can be used to determine whether the handoff is an intra- or inter-subnet handoff. Once $T2$ times out, the relay will be prohibited by the new AP and the old AP. Here we assume that some extension to DHCP for mobility support is implemented [18, 37], where the DHCP client in MS_i can detect the change of subnets. So, MS_i will actively execute the DHCP. After the link layer handoff, data is encrypted between MS_i and the new AP because a new data encryption key has been derived after the four-way handshake. On receiving the DHCP ACK message from the DHCP server of the new subnet (H11), the DHCP client will reconfigure MS_i 's IP address and network parameters. Moreover, this will trigger the new AP to stop relaying uplink data to the old AP and start directly relaying uplink data from MS_i to the Internet because a new IP is used (H12). For the old AP, the downlink data forwarding from CN to the new AP is continued until $T2$ times out. Because CN will still transmit data to the old subnet before the SIP mobility competes, this can help MS_i to continue to receive downlink data. After the DHCP, MS_i will execute the SIP

mobility procedure (H13). When the SIP mobility completes, CN will start to transmit data to the new subnet of MS_i . Then, the handoff procedure completes.

3.4 Experimental Results

In this section, we present some experimental results to verify the effectiveness of our proposed system and schemes. The evaluation includes two parts. First, we introduce our prototype of the proposed mobile broadband network, analyze its performance, compare its signaling cost to the MIPv6-NEMO scheme with routing optimization, and use voice calls applications to test the performance. Second, we estimate the performances of our proposed Cross-Layer Resource Management Mechanisms.

3.4.1 Evaluation of the Proposed Mobile Broadband Network

Our Prototype

We have implemented a prototype of the proposed system. In our prototype, the SIP-MNG is implemented over the Linux Fedora Core Release IV. The command *iptables* and the library *libipq* [40] are used to carry out the NAT traversal and SIP-Application Layer Gateway (ALG) (an application layer program to monitor and translate SIP messages). As to the external interfaces of SIP-MNG, we adopt the PHS WiWi Card MC-P300/P-Card MC-6550 and the Huawei E612 WCDMA PCMCIA card to connect to the PHS and WCDMA cellular networks, respectively. Our current push server is implemented by C++ over the Microsoft Windows XP. The push server contains a simple SIP stack and can support the SIP REFER method [58]. Both the SIP-MNG and the push server run a SMS agent to transmit/receive short messages. The communication protocol between the GSM interface and the SMS agent is achieved by the SMS AT commands [47]. To support multi-hop ad hoc routing, MSs in the MANET all run the Destination-Sequenced Distance-Vector (DSDV) protocol.

The testing environment is shown in Fig. 3.22. MSs are IBM X23 notebooks equipped with ASUS WL-167G USB2.0 WLAN adapters configured at the ad hoc mode. They run the Microsoft Windows XP with Windows messenger 5.1 as their SIP client software. The SIP-MNG is an IBM T42 notebook running the Linux Fedora Core Release IV; a Nokia card phone 2.0 is equipped to access GSM SMS. To access the Internet, the SIP-MNG is equipped with an embedded 802.11 WLAN chipset and a cellular interface (Huawei E612 WCDMA PCMCIA card/PHS J88 cellular phone). The PHS phone interface is driven by the SIP-MNG via a P-Card MC-6550 PCMCIA adaptor. The push server is an ASUS centrino notebook running the Microsoft Windows XP. It is also equipped with a Nokia

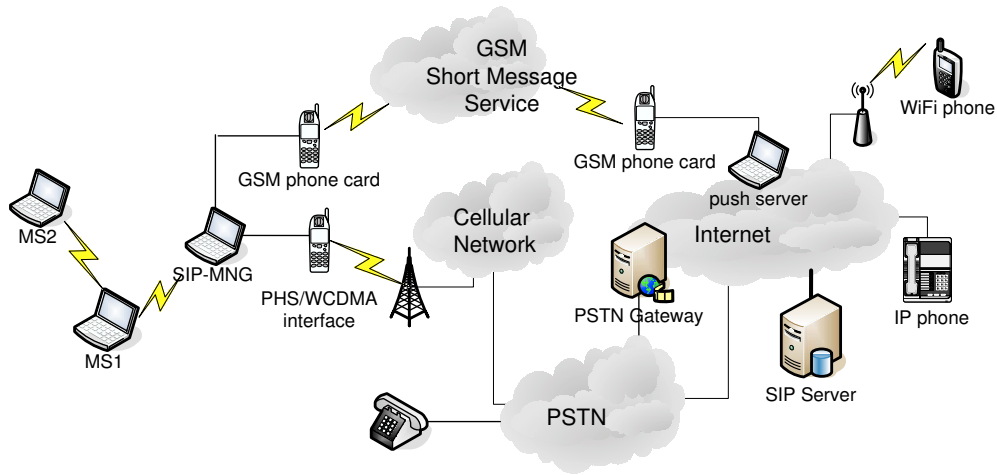


Figure 3.22: A testing environment of our SIP-based mobile broadband network system.

Table 3.5: Measurement of call setup time and capacities of different interfaces.

Item	external interface			
	PHS	WCDMA	GPRS	802.11b
call setup time	1.58 sec	1.44 sec	—	1.11 sec
maximum number of calls (with G.729)	2	5	0	12

card phone 2.0 to transmit short messages. An iptel SIP server [56] is used to serve as a server. In the Internet side, we have several SIP client terminals including D-Link IP phone and WiFi phone. All SIP clients inside the MANET or in the Internet use the SIP server as their home SIP server.

Call Setup Time and Maximum Number of Supported Calls

Based on the above environment, we have measured the call setup time and the maximum number of supported calls for different wireless interfaces. Some testing results are shown in Table 3.5. Each experiment here and in the following subsections was repeated 10 times. It shows that, to set up a call via a PHS J88 phone interface, it takes 1.58 seconds in average for MS2 to receive the ringing tone of a call from an IP phone in the Internet. If via the WCDMA PCMCIA card/embedded IEEE 802.11b interface, it requires 1.44/1.11 seconds in average, respectively. We do not provide results of the GPRS interface because in our experience a GPRS interface cannot provide enough bandwidth to support even one single voice call (GPRS downlink bandwidth is only 28.8kbps, uplink bandwidth is even less). The *call setup time* by cellular interfaces is longer than that by 802.11 interfaces because connecting MS2 and the IP phone via a cellular interface has to go through four networks: the Internet, PSTN, cellular network, and our MANET.

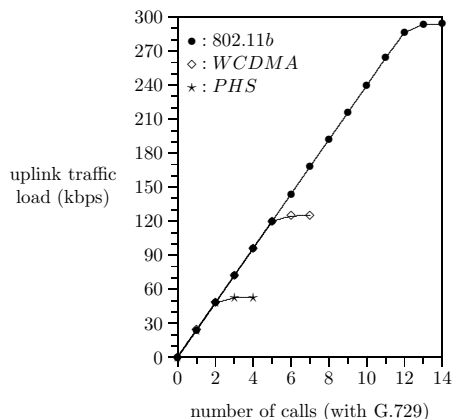


Figure 3.23: Uplink traffic load against the number of concurrent calls using one PHS/WCDMA/802.11b interface.

However, connecting them via a wireless interface only goes through two networks: the Internet and our MANET. The proposed system allows multiple calls to be supported by a single interface. With a PHS interface, the SIP-MNG can support two concurrent voice calls with acceptable quality ($< 1\%$ packet dropping rate), while with one WCDMA interface, it can support up to five concurrent voice calls. Interestingly, with an IEEE 802.11b wireless interface, the SIP-MNG can support up to 12 concurrent voice calls with G.729 as the codec.

We have also measured the uplink traffic load at the core network side for a PHS/WCDMA/802.11b interface. In this experiment, all calls from the mobile broadband network are destined to the same corresponding node in the Internet. Then, we monitor the traffic load on the corresponding node. We did not measure downlink load because the uplink bandwidth is lower than the downlink bandwidth in all existing cellular interfaces. Fig. 3.23 plots the uplink traffic load against the number of concurrent calls using a PHS/WCDMA/802.11b interface. As shown in Fig. 3.23, in the beginning, the uplink traffic load increases linearly as the number of calls increases. As the number of calls keeps increasing, the PHS's curve will saturate after there are more than two calls, the WCDMA's curve will saturate after there are more than five calls, and the IEEE 802.11b's curve will saturate after there are more than 12 calls. This gives the maximum number of calls that can be supported in Table 3.5. Our measured capacity for the 802.11b wireless interface is close to the analysis in [69], which claims that 11.4 G.729 calls can be supported in average. Note that, since we measure the traffic load at the core network side, the per call required bandwidth for any of the PHS/WCDMA/802.11b interfaces is the same (voice payload plus the TCP/IP/Ethernet overhead).

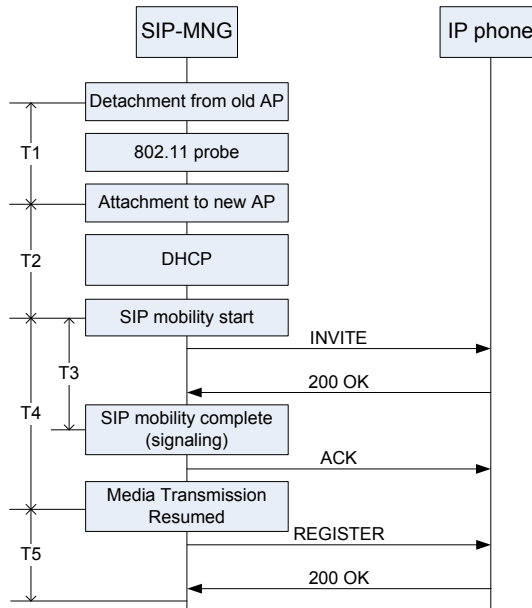


Figure 3.24: The handoff procedure when an IEEE 802.11 interface is used.

Table 3.6: Measurement of handoff latencies.

Item	external interface	
	Cellular (PHS)	Wireless (802.11b)
T3	742.87 ms	268.26 ms
T4	967.37 ms	279.54 ms
T5	267.69 ms	42.34 ms

Handoff Delay

Fig. 3.24 illustrates the handoff procedure when an IEEE 802.11 interface is used. The handoff delay can be divided into five parts: T1 (the delay of layer 2 handoff for an MS switching from one AP to another), T2 (the delay of layer 3 handoff for an MS getting a new IP in a new subnet), T3 (the delay of SIP mobility), T4 (the delay from the SIP-MNG activating the SIP mobility procedure to the media session being resumed), and T5 (the re-registration latency for the SIP-MNG to update MSs' contact information).

T1 and T2 are not the focus in our experiments because handoff rarely happens for cellular interfaces. In addition, lots of previous work [38, 6] have already provided empirical results of WLAN layer 2 and layer 3 handoff latencies. Therefore, we only evaluate T3, T4, and T5 for cellular and 802.11 interfaces. Table 3.6 shows the values of T3, T4, and T5 for different interfaces. For cellular interfaces, T4 consumes 967.37 ms in average, which is somewhat long. Fortunately, for cellular interfaces, handoff seldom happens because an operator's network normally covers quite a large area. For cellular interfaces, T5 is 267.69 ms in average. This is why we do re-invitation before re-registration when

Table 3.7: Latencies of the push mechanism.

Item	external interface		
	PHS (PHS J88)	WCDMA (Huawei E612)	802.11b (Centrino)
call setup time (with push mechanism)	23.03 sec	18.77 sec	23.11 sec
short message transmission time	5.31 sec	5.31 sec	5.31 sec
wireless interface reconnection time	13.05 sec	8.47 sec	14.05 sec

a handoff occurs, or the SIP mobility latency will be further increased by 270 ms. For 802.11 interfaces, T3 and T4 are 268.26 ms and 279.54 ms, respectively, in average. This is much shorter than the cellular interface case. The re-registration delay T5 for such interfaces is only 42.34 ms in average.

Performance of the Push Mechanism

By the proposed push mechanism, our system allows the external interfaces of the SIP-MNG to be disconnected when there is no Internet connection. In this part, we evaluate the call setup latency when the SIP-MNG is disconnected from the Internet with our push mechanism. Table 3.7 shows our testing results with different interfaces. If the interface is a PHS J88 phone (in the countryside areas), with our push mechanism, it costs 23.03 seconds in average for MH2 to receive the ringing tone from the IP phone. This latency consists of two major components: short message transmission time and wireless interface reconnection time (switch from the standby mode to the active mode). We found that the short message transmission time is 5.31 seconds in average and the wireless interface reconnection time takes 13.05 seconds in average. Replacing the PHS interface with a Huawei E612 WCDMA PCMCIA card, it shows that the call setup time can be reduced to 18.77 seconds in the disconnected case. To further divide the cost, Table 3.7 shows that the short message transmission time is also 5.31 seconds and the wireless interface reconnection time is 8.47 seconds in average. If the wireless interface is an IEEE 802.11 interface, with our push mechanism, it shows that the average call setup time is 23.11 seconds, within which the wireless interface reconnection time requires 14.05 seconds in average. The call setup time in the 802.11 case is longer than the cellular interfaces because reconnecting an 802.11 interface involves scanning for wireless networks and attaching to one of them, but the cellular interfaces are already in the standby mode. However, no matter which type of interfaces is used, the call setup time is not short. This is why we

design our push server to temporarily answer an incoming call to keep the session alive, or the caller may hang up the call before the call is established.

Comparison of Signaling Cost

Next, we compare the signaling cost of our proposed approach and the MIPv6-NEMO scheme with routing optimization. We consider two cases. One is the offline case, where SIP-MNG is disconnected from the Internet; the other is the online case, where SIP-MNG is connected to the Internet. In the offline case, for SIP-MNG, there is no SIP signaling cost when it moves from one network domain to another. The external push server would be SIP-MNG's representative, which answers and transfers incoming sessions for the mobile network. On the other hand, since the MIPv6-NEMO scheme does not support the disconnected feature, the MR still has to track network signaling and update with its HA when handoff. Suppose that the HA binding update cost is $cost_{HABU}$. In the online case, the SIP signaling cost of SIP-MNG when handoff will be $N \times cost_{SIP-reregistration} + S \times cost_{SIP-reINVITE}$, where N is the number of MSs in the mobile network, S is the number of sessions between the mobile network and the Internet. On the other hand, the MIPv6-NEMO scheme with routing optimization will require a cost of $cost_{HABU} + M \times cost_{BU}$ when handoff, here we assume that the routing optimization approach based on binding update for network prefixes is used, and M is the number of CNs that are communicating with the mobile network. To summarize, in the online case, the SIP has its disadvantage in terms of signaling cost as compared to the MIPv6-NEMO scheme with routing optimization. However, in the offline case, our approach incurs lower cost than the MIPv6-NEMO scheme, which may compensate some of the signaling cost in the on-line case.

3.4.2 Effectiveness of the Proposed Cross-Layer Resource Management Scheme

To verify the correctness and applicability of the proposed algorithm, an event-driven C++ simulator is developed. Unless otherwise stated, the following assumptions are made in our simulation. (1) The same call arrival model, call holding time, and call residence time as specified in Section 4 are used in the simulation. (2) Parameters of IEEE 802.11b and 802.11e are used. (3) We set TXOP limit to zero for four ACs, which means that a QSTA can only transmit one packet in each successful contention. (4) The communication channel is assumed to be error-free. (5) No RTS/CTS is used. (6) The BI=500 ms. (7) For AC_VO traffics, we set CWmin to 7, CWmax to 15, and AIFSN[AC_VO] to 2. For AC_BE traffics, we set CWmin=31, CWmax=1023, and AIFSN[AC_BE]=3. (8) Since

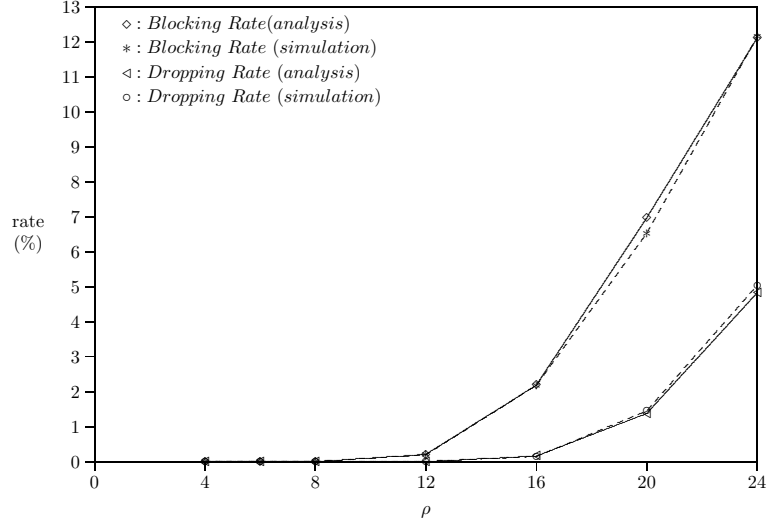


Figure 3.25: Comparison of simulation and analytical results on blocking rate and dropping rate ($\lambda_h = 0.8$, $\lambda_n = 1.2$).

QAP is more likely to be the performance bottleneck, we give it a higher priority by setting its TXOP[AC_VO] to N packets where N is the expected number of voice QSTAs in the QAP. (9) In QSTAs, each AC has a queue of size 50. (10) A voice packet will be dropped when it is buffered in a queue for more than 100 ms. (11) A packet can be re-transmitted at most 3 times. (12) G.726 with 32 kbps is used as the voice source. The offered network load is defined as $\rho = (\lambda_n + \lambda_h)/(\mu_r + \mu_h)$. To reach steady states, each simulation case is run with one million arrivals. The performance metrics are new call blocking rate, handoff call dropping rate, channel utilization, and the average voice packet dropping rate, P_d , where the channel utilization is defined as $\frac{1-B_{free}}{B_{total}} \times 100\%$, not the same as the channel utilization in [72]. The quality of a voice call is considered to be acceptable if its $P_d < 2\%$; otherwise, it is considered unacceptable.

Validation of Analytical Results

In this experiment, we assume that 40% of arrival calls are handoff calls. The channel occupancy time is 2 second. P_r is set to 0.8 when $B_{deg} \leq (B_{total} - B_{th})$. Fig. 3.25 shows the blocking rate and dropping rate of both analytical and simulation results. The maximal difference of blocking rate between simulation and analytical results is about 0.44%, which appears when $\rho = 20$. It can be seen that analytical results match well with simulation results. So, both our analytical and simulation results are correct and believable.

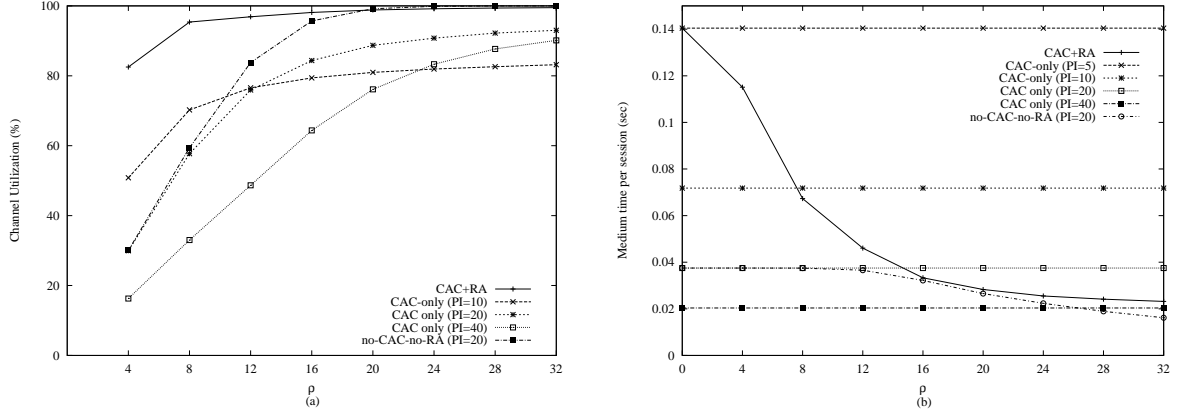


Figure 3.26: Comparisons of different schemes on: (a) channel utilization and (b) goodput.

Influence of CAC and RA

In this experiment, we want to evaluate the impact of CAC and RA. P_r is set to 0.8 when $B_{deg} \leq (B_{total} - B_{th})$. We compare our scheme against the CAC-only and “no-CAC-no-RA” cases. For the CAC-only case, the PIs of calls are fixed. Fig. 3.26(a) shows the channel utilization under different offered loads. Clearly, our scheme has very good utilization because calls can always be upgraded when there are extra resources. The no-CAC-no-RA (PI=20) case outperforms the CAC-only (PI=20) case because it accepts every incoming request in all network situations. Fig. 3.26(b) shows the medium time per session receives (which is approximated by the total used medium time divided by the total number of ongoing calls). With call admission control, the medium time of our scheme is better than that of the no-CAC-no-RA case. Even when the work load is high, our scheme can still guarantee the minimum bandwidth requirement of all calls. As $\rho \geq 32$, the minimum time of the no-CAC-no-RA case will drop to an unacceptable level. This shows that our scheme can well utilize network resources while guarantee the quality of calls.

Fig. 3.27 shows the average voice packet dropping rate, P_d , against different ρ for both CAC+RA and no-CAC-no-RA cases. Without CAC, the no-CAC-no-RA curve rapidly exceeds the 2% threshold, which means an unacceptable voice quality. With CAC, our algorithm has a very small P_d because the resource usage is well under control. This result shows that CAC is definitely necessary, or the quality of real-time services will easily become unacceptable.

Fig. 3.28 shows the new call blocking rate and handoff call dropping rate versus different offered loads. The rates of the no-CAC-no-RA case are all zero because every incoming request is accepted. From Fig. 3.28(a), we see that our scheme is only slightly worse than the CAC-only (PI=40) case after $\rho \geq 12$ because of our call acceptance policy. However, the benefit is our lower handoff call dropping rate, as shown in Fig. 3.28(b).

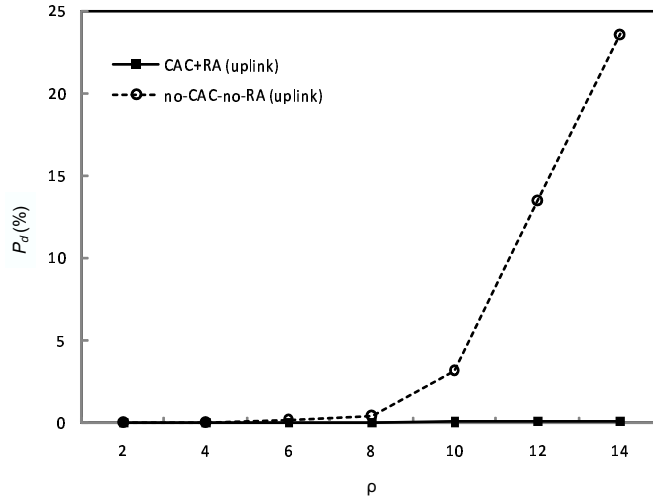


Figure 3.27: Comparison of different schemes on their average voice packet dropping rates (P_d).

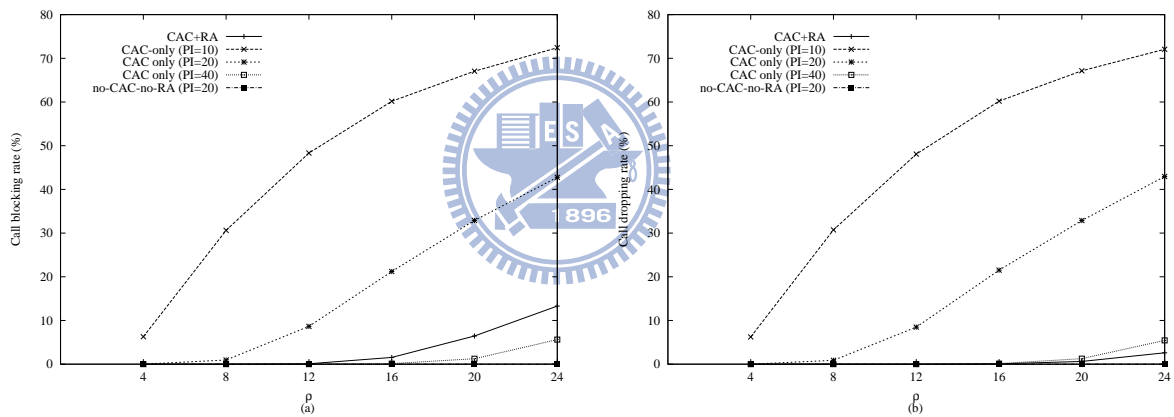


Figure 3.28: Comparisons of: (a) call blocking rate and (b) call dropping rate.

Influence of P_r

The value of P_r reflects the possibility that a QAP permits new calls to start. Clearly, a larger P_r will benefit new calls but hurt handoff calls. Fig. 3.29 shows the impact of P_r on call blocking and dropping probabilities. From these curves, a suggested value of P_r could range from 0.2 to 0.6.

Influence of Traffic Characteristic

Next, we evaluate the influence of traffic characteristic. We change the percentage of handoff calls while keep the offered load unchanged. Fig. 3.30 shows this impact on call blocking and call dropping rates. We can see that our scheme is quite insensitive to this

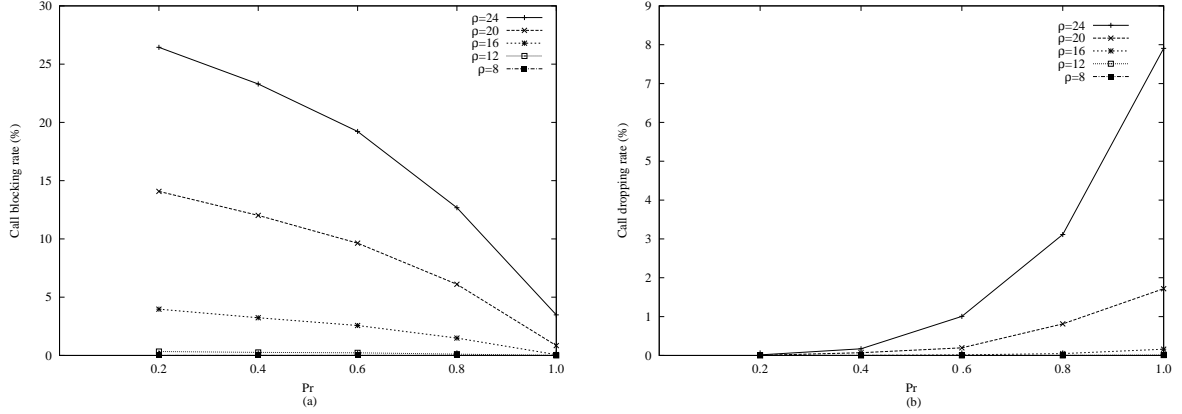


Figure 3.29: The impact of P_r on: (a) call blocking rate and (b) call dropping rate.

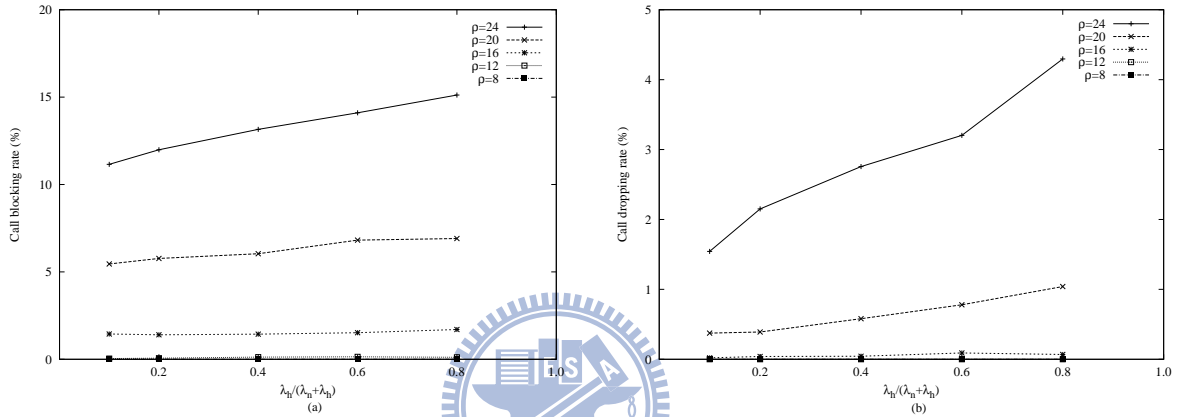


Figure 3.30: The impact of the percentage of handoff calls on: (a) call blocking rate and (b) call dropping rate.

change, unless the offered load is very high. This concludes that our scheme can provide good QoS to handoff calls.

Influence of Additional BE Flows

The above experiments all assume that VoIP is the only traffic in the network. In this experiment, we add some additional static QSTAs, each generating best-effort data (AC_BE) with a rate of 480kbps to compete with VoIP traffics. As we can see in Fig. 3.31, if AIFSN[AC_BE] is too small (such as 3), the AC_BE traffics will significantly affect the performance of VoIP traffics. When the voice load is high (such as $\rho = 32$), the value of P_d is much too high. Even in the light load case ($\rho = 4$), we see $P_d > 2\%$ when there are more than 20 BE streams. The effect can be reduced by enlarging the value of AIFSN[AC_BE]. As shown in Fig. 3.31, with a slightly larger AIFSN for AC_BE, the dropping rates for voice packets are significantly reduced. When $\rho = 32$ and AIFSN[AC_BE]=15, $P_d = 0.50\%$ and 0.53% for BE_QSTA=2 and 8, respectively, which are improved by 89.46% and 97%, respectively, as opposed to AIFSN[AC_BE]=3. This also implies that AIFSN can effec-

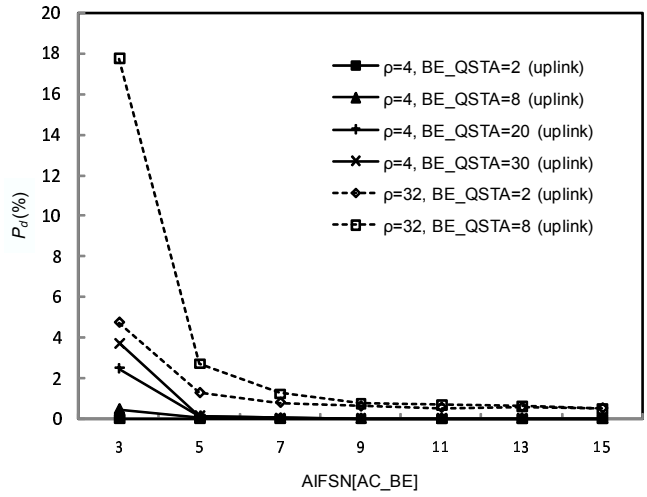
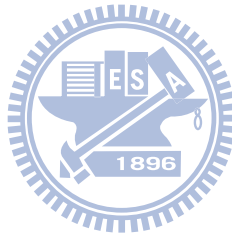


Figure 3.31: Impact of interference from AC_BE traffics with various AIFSNs for AC_BE traffics.

tively help differentiate the priorities of voice and best-effort packets.



Chapter 4

Power Saving Class Management for Single MS in Mobile Broadband Networks

IEEE 802.16/WiMAX [25] has been considered as a promising approach for supporting broadband wireless access. The IEEE 802.16e standard [26] defines the mobility support for mobile stations (MSs). Similar to most wireless systems, conserving energy is a critical issue for MSs. In IEEE 802.16e, three types of *Power Saving Classes (PSCs)* are defined to meet different traffic characteristics. Each PSC consists of a sequence of interleaved listening and sleep windows, and can support one or multiple traffic flows in an MS with similar characteristics. An MS can turn off its radio interface when all its PSCs are in their sleep windows, but has to wake up when any PSC is in a listening window. However, from MSs' point of view, the standard does not define how multiple flows should be put into one PSC and how multiple PSCs of an MS should cooperate with each other for better energy efficiency. At the same time, it needs to answer how to determine the parameters of each PSC, such as start frame, listening window size, and sleep window size, and how to guarantee QoS of traffic flows when multiple PSCs coexist. There have been some works focusing on the PSC management problem over IEEE 802.16e wireless networks. For single MS case, [61, 64] propose to apply one single PSC to serve all connections. However, these schemes only consider the strictest delay bound of connections. This could incur waste of bandwidth and extra listening windows. In this chapter, focusing on single MS (a BS-MS pair), we propose to assign PSCs to connections according to their QoS characteristics. Then, simulation results of our proposed schemes are shown in the end of the chapter.

4.1 Motivation and Problem Definition

We first motivate our work and then present our problem definition. For PSCs of type II, previous works [61, 64] try to form one single PSC to serve all connections in an MS. While this is simpler, there are several drawbacks. First, the strictest delay bound among all connections must be followed to avoid missing any deadline. Second, in each listening window, the BS needs to reserve the maximum bandwidth to accommodate the maximum possible load. This leads to waste of bandwidth and extra awake frames. If each connection has its own PSC according to its packet inter-arrival time and delay bound, the MS only needs to stay awake at proper times, thus incurring less resource waste. Fig. 4.1 shows an example with two connections C_1 and C_2 , which have packet inter-arrival time of $PI_1 = 3F$ and $PI_2 = 6F$, delay bounds of $D_1 = 6F$ and $D_2 = 18F$, and expected packet sizes of $S_1 = \frac{2}{5}B$ and $S_2 = \frac{2}{5}B$, respectively, where F is the frame duration and B is the maximum available resource per frame for the MS. As Fig. 4.1(a) shows, if the *periodic on-off scheme (PS)* [61] is applied, there is only one PSC of type II, which has a sleeping cycle of 6 frames (i.e., the strictest delay bound $6F$ by C_1) and a listening window is 2 frames per cycle (i.e., the traffic demand of C_1 and C_2 per $6F$, $\lceil \frac{6F}{3F} \rceil S_1 + \lceil \frac{6F}{6F} \rceil S_2 = \frac{6}{5}B$, after taking the ceiling function). As Fig. 4.1(b) shows, by using two PSCs, we can construct a PSC P_1 of period $T_1 = 6$ frames for C_1 and a PSC P_2 of period $T_2 = 18$ frames for C_2 . P_1 and P_2 need 1 and 2 frames of listening windows per cycle, respectively (note that two PSCs can share some active frames as long as their traffics can be consumed). Compared to Fig. 4.1(a), which needs 2 listening frames per 6 frames, Fig. 4.1(b) needs 4 listening frames per 18 frames, thus reducing the energy cost by 33.3%.

We consider the problem of sleep scheduling for single MS as follows. We are given n real-time connections C_i , $i = 1..n$ between an MS and a BS. Each connection C_i has an expected packet inter-arrival time PI_i (ms), a delay bound for packets D_i (ms), and an expected packet size S_i (bits). Assuming that the BS can allocate to the MS at most B (bits) per frame, the goal is to assign PSCs to connections, where each PSC P_i is with sleeping cycle T_i (frames), listening window T_i^L (frames), sleep window T_i^S (frame), and starting frame of the first listening window T_i^f , such that the ratio of active frames for the MS is minimized and the QoS of each connection can be guaranteed.

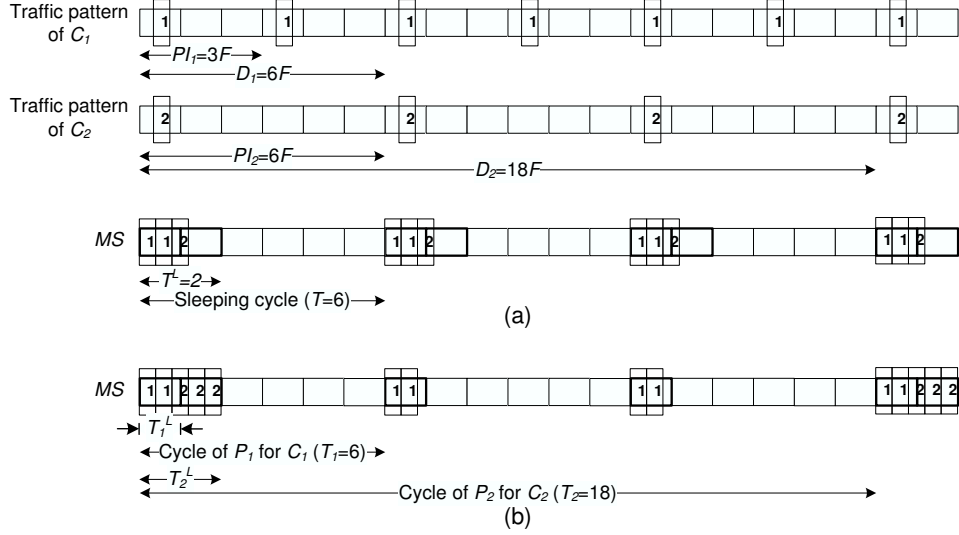


Figure 4.1: Sleep scheduling for two connections C_1 and C_2 using (a) one PSC and (b) two PSCs.

4.2 Fold-and-Demultiplex (FD) Method with QoS-Guaranteed Packet Scheduler

We consider an MS with M real-time uplink connections, N real-time downlink connections, and U non-real-time connections, where each real-time connection has a minimum reserved traffic rate $C_i.MRTR$ (bits/sec). The method consists of four steps. First, only real-time connections are considered and *one* tentative PSC of type II per connection is created. Each of such PSCs has a wake-up period that is an integer multiple of that with the shortest period. Second, we fold all these PSCs into one so as to calculate the overall bandwidth requirement. Third, we demultiplex the above result into multiple real PSCs. The last step will include non-real-time connections.

4.2.1 Creating Tentative PSCs

In this step, only real-time connections are considered. It will compute one tentative PSC for each connection and send the result to the BS via a MOB_SLP-REQ message. It includes three steps: 1) For each C_i , $i = 1..M + N$, create a tentative PSC of type II according to its QoS parameters. 2) To increase the overlapping of listening windows, adjust these PSCs such that their wake-up periods are integer multiples of the smallest one. 3) Adjust these connections' QoS parameters to adapt to their sleep behaviors.

1. For each C_i , $i = 1..M + N$, we define a tentative PSC P_i^{II} as follows:

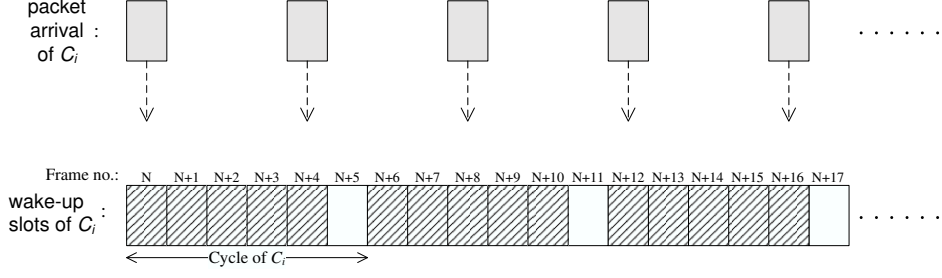


Figure 4.2: A mismatch example between the packet arrival time of a flow and its wake-up period.

$$P_i^{II}.T_L = \begin{cases} 2, & \text{if } C_i \text{ is of type uplink RT-VR} \\ 1, & \text{otherwise} \end{cases}, \quad (4.1)$$

$$P_i^{II}.T_i = \left\lfloor \frac{C_i.D_i}{2 \times F} \right\rfloor. \quad (4.2)$$

Here F is the length of a frame. For an uplink RT-VR connection, since it requires an additional frame to send its bandwidth request to the BS, its listening window should be two frames. For other types, we assume that one frame is sufficient (later on, we will relax this). We regard T_i as the *wake-up period* of a PSC. Eq. (4.2) ensures that C_i 's wake-up period is no more than $\frac{1}{2}$ of its maximum delay $C_i.D_i$. Such setting guarantees bounded delay for each packet of C_i (to be explained later).

2. We further adjust these PSCs to increase their overlapping. Let $p_{min} = \min \{P_i^{II}.T_i \mid i = 1..M + N\}$, i.e., the smallest wake-up period. We adjust each C_i 's wake-up period as follows:

$$P_i^{II}.T_i = \left\lceil \frac{P_i^{II}.T_i}{p_{min}} \right\rceil \times p_{min}. \quad (4.3)$$

This makes C_i 's wake-up period an integer multiple of p_{min} . For example, if there are three PSCs with $P_1^{II}.T_L = 1$, $P_1^{II}.T_i = 4$, $P_2^{II}.T_L = 1$, $P_2^{II}.T_i = 5$, $P_3^{II}.T_L = 1$, and $P_3^{II}.T_i = 9$, then $p_{min} = 4$. After adjustment, $P_2^{II}.T_i = 4$ and $P_3^{II}.T_i = 8$. Before the adjustment, there are 84 listening frames per 180 frames. After the adjustment, there is 1 listening frame per 4 frames.

3. Because of the MS's sleeping behavior, a flow may need to deliver more traffic during a listening window. So changing its QoS parameters may be needed. Consider the connection in Fig. 4.2, where the packet inter-arrival time is 4 frames but its wake-up period is 6 frames. A listening window needs to serve one or two packets each

time. We need to reserve sufficient bandwidth in each listening window to serve the demand (in this example, it is the size of two packets). Therefore, we suggest to change the $MRTR$ of each C_i as follows:

$$\begin{aligned}
C_i.MRTR &= [(\text{max. no. of arrivals per wake-up period}) \times (\text{expected size per arrival})] \\
&\quad \times (\text{no. of wake-up periods per second}) \\
&= \left[\left\lceil \frac{P_i^{II}.T_i \times F}{C_i.PI_i} \right\rceil \times \left(C_i.MRTR \times \frac{C_i.PI_i}{1000} \right) \right] \times \left(\frac{1000}{P_i^{II}.T_i \times F} \right) \\
&= \left\lceil \frac{P_i^{II}.T_i \times F}{C_i.PI_i} \right\rceil \times \frac{C_i.MRTR \times C_i.PI_i}{P_i^{II}.T_i \times F}. \tag{4.4}
\end{aligned}$$

Since $P_i^{II}.T_S$ has changed, C_i 's grant interval/polling interval should be changed to $P_i^{II}.T_i \times F$. These changes can be updated by DSC-REQ messages. The effect is a slight increase of bandwidth demand with reduction the buffering delay. In Fig. 4.2, we will allocate space to deliver $\lceil \frac{6}{4} \rceil = 2$ packets per listening window, so that the new $MRTR = 2 \times \frac{C_i.MRTR \times 20}{6F} = \frac{4}{3} \times C_i.MRTR$ and the new grant interval/polling interval is $6F$.

4.2.2 Folding PSCs into a State Series

Next, we will fold the above PSCs into an infinite periodical series of real numbers standing for bandwidth requirements. Then depending on the available bandwidth per frame, the real series is converted into a binary series, standing for the active or sleeping state of each frame. Note that the series does not meet the definition of PSC. Later on, we will demultiplex it into actual PSCs. For ease of presentation, we define T_i as C_i 's wake-up period and $p_{lcm} = lcm\{T_i \mid i = 1..M + N\}$ (the least common multiplier of T_i s).

We first define a series for each C_i . For C_i of type UGS/ERT-VR, due to its sleeping behavior, the amount of data b_i that it has to transmit during a listening window is

$$b_i = C_i.MRTR \times P_i^{II}.T_i \times F. \tag{4.5}$$

Assuming frame 0 to be a listening window, C_i 's bandwidth requirement in each frame can be represented by a periodical series $S_i(t), t = 0..\infty$, where $k \in N$:

$$S_i(k \times T_i + t) = \begin{cases} b_i, & t = 0 \\ 0, & \text{otherwise} \end{cases}.$$

For C_i of type RT-VR, in each listening window, it requires a fixed bandwidth Δb (to submit its request) in the first frame and, if needed, a bandwidth of b_i in the second frame. So, its bandwidth requirement can be represented by

$$S_i(k \times T_i + t) = \begin{cases} \Delta b, & t = 0 \\ b_i, & t = 1 \\ 0, & \text{otherwise} \end{cases} .$$

Putting all these together, the bandwidth requirement of the MS in each frame can be written as a series:

$$\hat{S}(t) = \sum_{t \geq 0} S_i(t) .$$

Note that the above discussion does not distinguish uplink from downlink communications. In fact, $\hat{S}(t)$ should be separated into two series, one for uplink and one for downlink. With this understanding, we will still use $\hat{S}(t)$ for simplicity. Since each $S_i(t)$, $i = 1..M + N$, has a period of T_i , it also has a period of p_{lcm} . It follows that the summation of them also has a period of p_{lcm} .

Lemma 4.2.1. $\hat{S}(t)$ is a periodical series with period p_{lcm} .

Next, we convert the real series $\hat{S}(t)$ to a binary state series $\tilde{S}(t)$, where 1 and 0 mean active and sleeping states, respectively:

$$\tilde{S}(t) = \begin{cases} 1, & \text{if } a(t) > 0 \\ 0, & \text{otherwise} \end{cases} , \quad (4.6)$$

where $a(t) = \min \left\{ \sum_{k=0}^t \hat{S}(k) - \sum_{k=0}^{t-1} a(k), B \right\}$ is the bandwidth to be allocated to the MS in the t -th frame and B is the maximum bandwidth that can be allocated to the MS in a frame (this is decided by the BS). Intuitively, $\sum_{k=0}^t \hat{S}(k)$ is the total bandwidth required up to the t -th frame and $\sum_{k=0}^{t-1} a(k)$ is the actual bandwidth consumed by the MS up to the $(t-1)$ -th frame. So the first term in $\min()$ is the actual bandwidth required in the t -th frame; however, the actual allocation should be bounded by B . If $a(t) > 0$, the MS should be active in the t -th frame, enforcing $\tilde{S}(t) = 1$; otherwise, it can go to sleep, making $\tilde{S}(t) = 0$.

Figure 4.3 is an example. Fig. 4.3(a) shows the total bandwidth requirement per frame from three connections (i.e., $\hat{S}(t)$). Fig. 4.3(b) shows the actual resource allocation in each frame (i.e., $a(t)$). Fig. 4.3(c) is the state series $\tilde{S}(t)$ of the MS.

Lemma 4.2.2. $\tilde{S}(t)$ is a periodical binary series with period p_{lcm} .

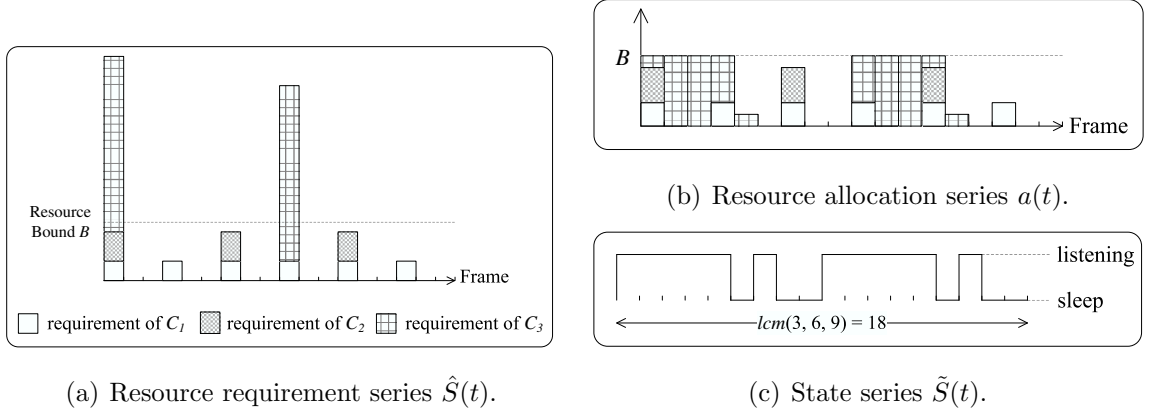


Figure 4.3: Example of the state series construction.

Proof. $\tilde{S}(t)$ is a binary series which alternates between continuous 1s and continuous 0s infinitely. Consider each group of continuous 1s in $\tilde{S}(t)$; let it start from position t_{st} and end at position t_{end} . Let us define C as the set of connections, each of which has a non-zero bandwidth requirement during the interval $[t_{st} : t_{end}]$, i.e., $C = \left\{ C_i \mid \sum_{j=t_{st}}^{t_{end}} S_i(j) > 0 \right\}$. Intuitively, each connection in C contributes to the wake-up behavior (continuous 1s) during the interval $[t_{st} : t_{end}]$. Since each $S_i(j)$ corresponding to $C_i \in S_c$ is periodical, the same bandwidth requirements from C occurring during the interval $[t_{st} : t_{end}]$ must occur again in interval $[t_{st} + i \times p_{lcm} : t_{end} + i \times p_{lcm}]$, where $i \in \mathbb{Z}$. Therefore, $\tilde{S}(t)$ must contain all 1s in interval $[t_{st} + i \times p_{lcm}, t_{end} + i \times p_{lcm}]$ for each $i \in \mathbb{Z}$. It follows that the theorem is true. \square

4.2.3 Demultiplexing the State Series into PSCs

Although $\tilde{S}(t)$ is periodical (Lemma 4.2.2), if we look at any subsequence of length p_{lcm} in $\tilde{S}(t)$, it may contain multiple groups of continuous 1s interleaved by 0s and thus does not fit into the definition of PSC. Below, we show how to demultiplex $\tilde{S}(t)$ into multiple state series, each meeting the definition of PSC. By Lemma 4.2.2, a straightforward approach is to pick the first p_{lcm} bits of $\tilde{S}(t)$ and let each group of continuous 1s be a PSC (this is in fact how the proof of Lemma 4.2.2 works). However, this may generate too many PSCs. Still, using the first p_{lcm} bits of $\tilde{S}(t)$, we propose a scheme based on folding $\tilde{S}(t)$ to put duplicate continuous 1s together. This may result in less PSCs.

1. Denote the first p_{lcm} bits of $\tilde{S}(t)$ by $\tilde{S}[0 : p_{lcm} - 1]$. Our goal is to construct a set C of PSCs. Initially, let $C = \emptyset$.
2. For $i = 1$ to $\frac{p_{lcm}}{p_{min}}$ do
 - (a) If p_{lcm} is not divisible by $i \times p_{min}$, skip this i and go back to step 2. Otherwise, cut $\tilde{S}[0 : p_{lcm} - 1]$ into $\frac{p_{lcm}}{i \times p_{min}}$ segments, each of length $i \times p_{min}$ and then

fold them together by the bitwise-AND operator into a $(i \times p_{min})$ -bit string $T[0 : i \times p_{min} - 1]$, i.e.,

$$T[j] = \tilde{S}(j) \wedge \tilde{S}(j + i \times p_{min}) \wedge \tilde{S}(j + 2i \times p_{min}) \wedge \cdots ,$$

for $j = 0..i \times p_{min} - 1$.

- (b) Scan string $T[0 : i \times p_{min} - 1]$ from left to right and consider each group of 1s in the string. Suppose that there is a group of 1s starting from the x -th bit to the y -th bit. Let $T'[0 : i \times p_{min} - 1]$ be a binary string which has all 1s from the x -th bit to the y -th bit and all 0s in the other places. We try to form a PSC from $T'[0 : i \times p_{min} - 1]$ as follows.

- i. Check if $T'[0 : i \times p_{min} - 1]$ is redundant by calling the procedure $Check_Redundancy(C, T'[0 : i \times p_{min} - 1])$.
- ii. If the response is negative (i.e., not redundant), then add the string $T'[0 : i \times p_{min} - 1]$ to C .

3. For each string str in C , we generate a PSC of type II. Let str have 1s from the x -th bit to the y -th bit. The PSC can be defined by setting $T_L = y - x + 1$ and $T_S = |str| - T_L$, where T_L and T_S are the sizes of listening window and sleep window, respectively.

Procedure $Check_Redundancy()$ is shown below. The binary string $Tested_Str$, which stands for a potential PSC, is redundant if each of its 1s already appears in a PSC in C . The PSCs in C are converted to a string $W[0 : p_{lcm} - 1]$ by “bitwise-OR” the strings of all PSCs. Similarly, string $Tested_Str$ is converted to a string $W'[0 : p_{lcm} - 1]$. Then we compare $W[]$ and $W'[]$ to see if $Tested_Str$ is redundant.

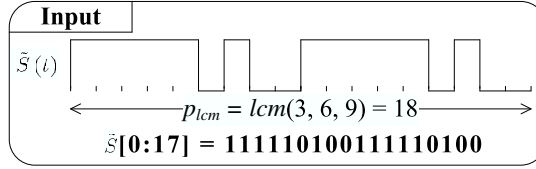
Procedure Check_Redundancy(C, Tested_Str):

1. Let $W[0 : p_{lcm} - 1]$ and $W'[0 : p_{lcm} - 1]$ be two binary arrays. For $j = 0$ to $p_{lcm} - 1$, we define

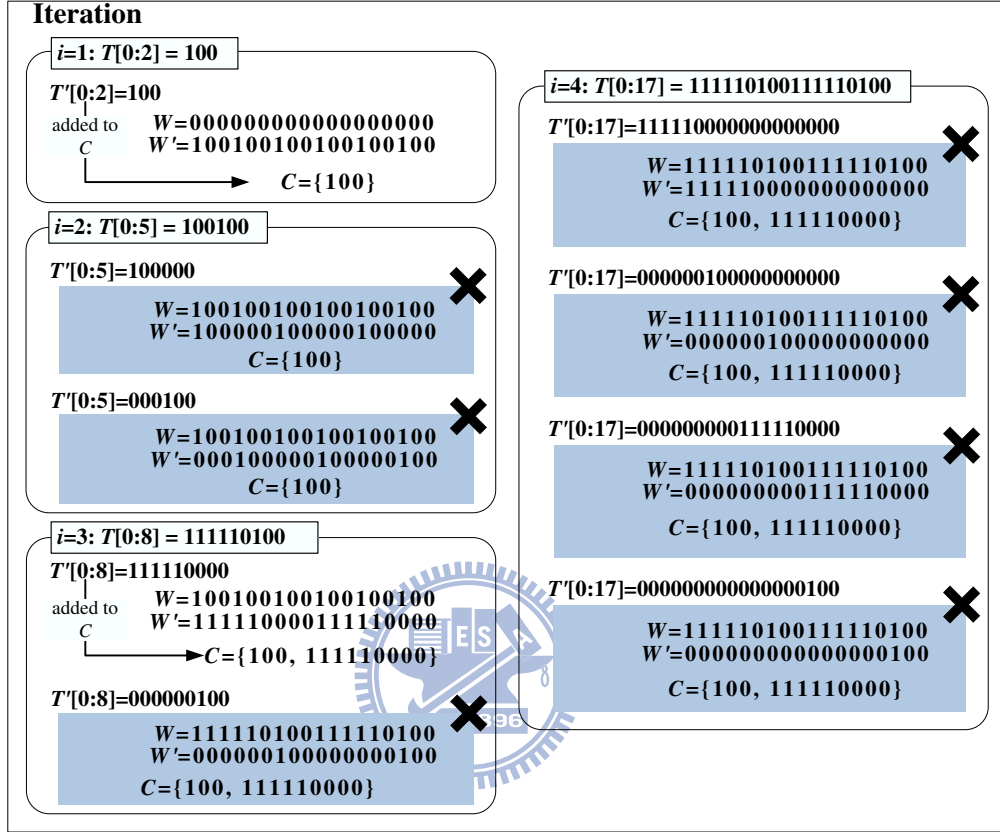
$$lCIW[j] = \bigvee_{Str \in C} Str[j \bmod |Str|]$$

$$W'[j] = \begin{cases} 1, & \text{if } Tested_Str[j \bmod |Tested_Str|] = 1 \\ 0, & \text{otherwise} \end{cases} .$$

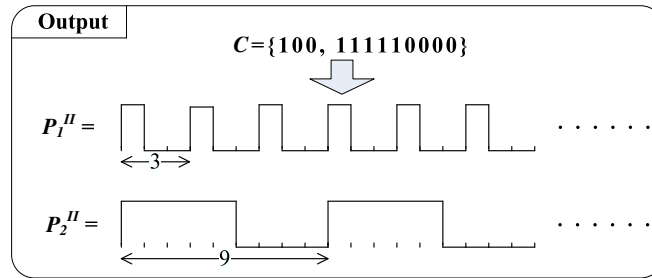
2. If $W'[j] = 1$ implies $W[j] = 1$ for all $j = 0..p_{lcm} - 1$, then a “positive” response is returned; otherwise, a “negative” response is returned.



(a)



(b)



(c)

Figure 4.4: Example of the PSC demultiplexing procedure.

The new definitions of PSCs can be notified to the MS by MOB_SLP-REQs. For example, Fig. 4.4(a) shows a state series $\tilde{S}(t)$ with $p_{lcm} = 18$ and $p_{min} = 3$. In Fig. 4.4(b), $\tilde{S}(t)$ is cut into segments, each of 3 bits. By “bitwise-AND” these segments, we get a string $T[0 : 2] = 100$. So, we add a PSC of type II with $T_L = 1$ and $T_S = 2$ to C . In the next iteration, $\tilde{S}(t)$ is cut into segments, each of 6 bits. By “bitwise-AND” these segments,

we have $T[0 : 5] = 100100$. Then we retrieve two T' strings, 100000 and 000100, and call *Check_Redundancy()* twice. Both strings are redundant because their active patterns have already been covered by string 100. Next, $\tilde{S}(t)$ is cut into segments, each of 9 bits. A binary string $T[0 : 8] = 111110100$ is obtained, from which two T' strings, 111110000 and 000000100, can be retrieved. The former is not redundant but the latter is, so one more PSC of type II with $T_L = 5$ and $T_S = 4$ is added to C . In the last iteration, the string $\tilde{S}[0 : p_{lem} - 1]$ does not add any new PSC. The two final PSCs are shown in Fig. 4.4(c).

4.2.4 Including a PSC of Type I for Non-real-time Connections

For non-real-time connections, we will define only one PSC of type I, P^I , for all of them. Whenever P^I enters a listening window, the BS will send the MS a broadcasting MOB-TRF-IND message containing a bitmap to indicate whether there are packets buffered at the BS. If there are packets for the MS, P^I will be deactivated until all packets are transmitted. Then P^I will be re-activated from the initial sleep window. In this work, we do not concern non-real-time connections' QoS and such traffics have the lowest priority.

Recall that we already construct a set C of PSCs of type II with a basic wake-up period of p_{min} frames. Assuming that P^I is more likely to enter the maximum sleep window of T_{S_max} (considering that these are non-real-time traffics), we will try to schedule the re-activation time of P^I such that the periodical (maximum) listening windows of P^I will align with the listening windows controlled by p_{min} . More specifically, we will tune the parameters of P^I such that $P^I.T_{S_max} + P^I.T_L$ is an integer multiple of p_{min} .

The problem is formulated as follows. We are given the initial values of T_L , T_{S_init} , and T_{S_max} . We assume that $T_{S_max} \geq p_{min}$ (it is unlikely that $T_{S_max} < p_{min}$ considering that these are non-real-time traffics). Without loss of generality, assume that the PSCs in C have common active frames appearing at integer multiples of p_{min} and at frame t the MS intends to reactivate its P^I . Our goal is to find a waiting interval Δt and modified parameters T'_{S_init} , T'_{S_max} , and T'_L such that P^I will actually enter its sleep window at frame $(t + \Delta t)$ and when P^I 's sleep window size reaches T'_{S_max} , its listening windows will appear at frame numbers that are integer multiples of p_{min} . To achieve this goal, we first set

$$T'_{S_init} = T_{S_init} \quad (4.7)$$

$$T'_L = 1 \quad (4.8)$$

$$T'_{S_max} = \left\lfloor \frac{T_{S_max}}{p_{min}} \right\rfloor \times p_{min} - T'_L. \quad (4.9)$$

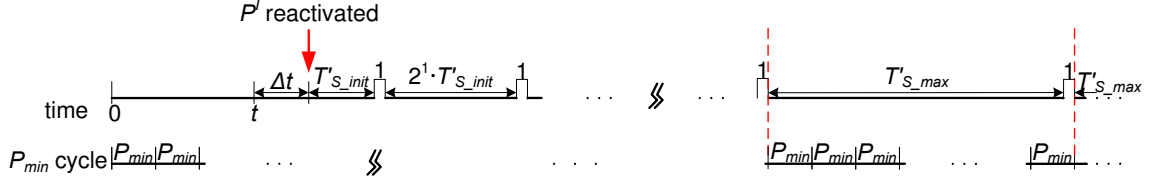


Figure 4.5: Inserting a waiting interval Δt such that after P^I reaching $T'_{S_{max}}$, the listening windows of P^I will align with the listening windows of PSCs in C .

So $T'_{S_{max}} + T'_L$ is divisible by p_{min} . Now, if there is no packet arrival, P^I will enter its first sleep window at frame $(t + \Delta t)$, sleep for $T'_{S_{init}}$ frames, wake up for 1 frame, sleep for $2 \times T'_{S_{init}}$ frames, wake up for 1 frame, sleep for $4 \times T'_{S_{init}}$ frames, \dots , until reaching the maximum sleep window size of $T'_{S_{max}}$. If so, the first listening window after the first maximum sleep window will appear at frame number

$$t_{flw} = t + P^I \cdot T'_{S_{init}} + 1 + 2 \times P^I \cdot T'_{S_{init}} + 1 + \dots + P^I \cdot T'_{S_{max}} + 1. \quad (4.10)$$

So Δt can be set to the smallest number such that $t_{flw} + \Delta t$ is divisible by p_{min} . The above concept is illustrated in Fig. 4.5.

An alternative is to not always start with $T'_{S_{init}}$, but instead start with a larger $2^j \times T'_{S_{init}}$ for some j . We can easily rewrite Eq.(4.10) to identify a new $\Delta t'$ with the same alignment property.

4.2.5 QoS-Guaranteed Packet Scheduler for FD Method

The above fold-and-demultiplex method can form a set C of PSCs that reserve sufficient bandwidths for connections while achieve energy efficiency. However, when packets from multiple connections arrive at the same time, it is still unclear how to schedule their transmissions to ensure their delay constraints. In this section, we propose a packet scheduler that can guarantee bounded delays for packets under the fold-and-demultiplex method.

Recall that our method will create two tentative series $\hat{S}(t)$ and $\tilde{S}(t)$ to represent the MS's bandwidth requirement and listening windows. We will prove our result through these two series. Consider the $\hat{S}(t)$ in Fig. 4.6. Suppose a packet $Data_i$ of connection C_i arriving at frame t_a . To analyze the delay that $Data_i$ may experience, consider C_i 's bandwidth requirement series $S_i(t)$. Let t_b be the first frame after t_a such that $S_i(t_b) \neq 0$. Conceptually, an amount of $S_i(t_b)$ bandwidth will be allocated to C_i at frame t_b to deliver $Data_i$ (refer to Fig. 4.6). However, it is clear from our method that the aggregated bandwidth requirement $\hat{S}(t_b)$ may exceed the capacity B , so it may take several frames

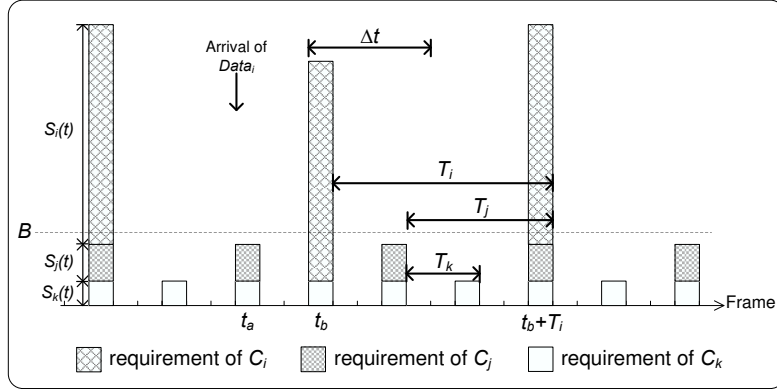


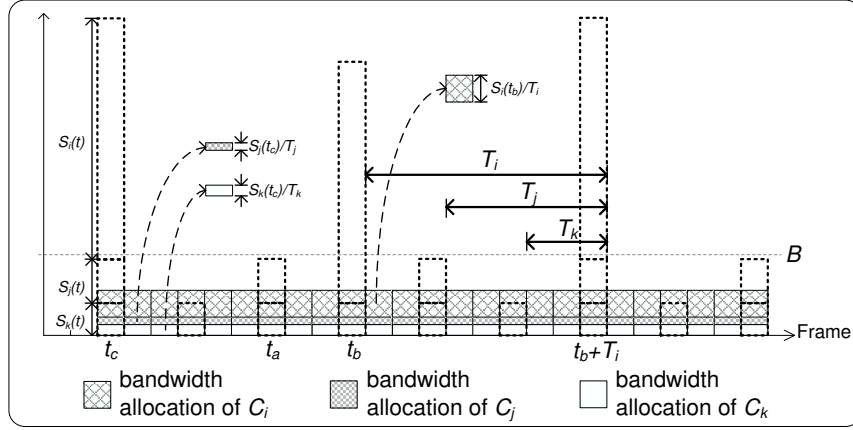
Figure 4.6: Illustration of packet delay analysis.

to serve $S_i(t_b)$. Let Δt be the number of frames that the request $S_i(t_b)$ is served. Then the total delay experienced by $Data_i$ is $((t_b - t_a) + \Delta t)$ frames. The first part $(t_b - t_a)$ can be bounded, while the second part (Δt) actually depends on the scheduler. Below, we will propose a scheduler that ensures $((t_b - t_a) + \Delta t)F \leq C_i \cdot D_{max}$.

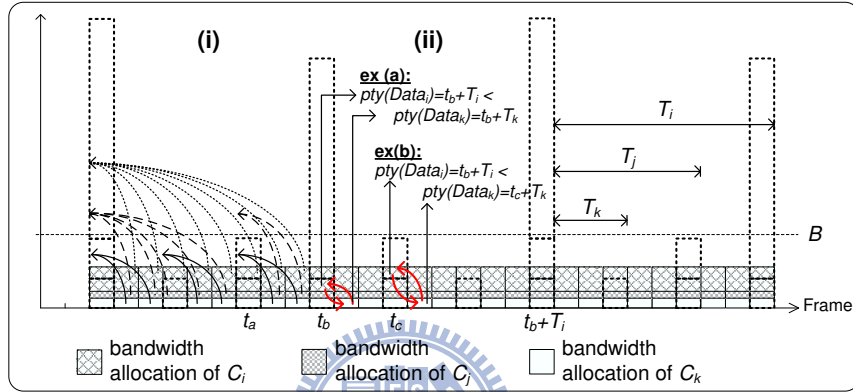
The proposed scheduler schedules data for transmission following an *earliest-next-bandwidth-first* policy. For any $Data_i$ of C_i arriving at frame t_a , let t_b be the first frame after t_a such that $S_i(t_b) > 0$. We assign a priority $pty(Data_i) = t_b + T_i$ to $Data_i$. Note that $t_b + T_i$ is the next frame such that $S_i(t_b + T_i) > 0$. Here a lower number means a higher priority. Then the scheduler simply schedules data for transmission in each active frame according to their priorities. Such a scheduling guarantees that $Data_i$ can be completely delivered before frame $t_b + T_i$. Combining the following theorem and the fact that $C_i \cdot D_{max} \geq 2T_i \times F$ (refer to Eq. (4.2)), our scheme ensures each packet's delay bound.

Theorem 4.2.3. *Under the fold-and-demultiplex method, the earliest-next-bandwidth-first policy guarantees that if data all arrives as planned, then each packet's delay is bounded by $2T_i$ frames, where T_i is the wake-up period of the corresponding PSC P_i^{II} .*

Proof. We use the scenario in Fig. 4.6 to develop our proof. Since t_b is the first frame after t_a such that $S_i(t_b) > 0$, it is clear that $t_b - t_a \leq T_i$. So we only need to prove that $\Delta t \leq T_i$. We develop an “imaginary” scheduling that guarantees $\Delta t \leq T_i$ and then show that our policy can not do worse than it, so our policy also ensures $\Delta t \leq T_i$. The imaginary scheduler assumes that data is infinitely divisible, so all data of C_i associated to $S_i(t_b)$ can be evenly served by frames $t_b, t_b + 1, \dots, t_b + T_i - 1$. For example, Fig. 4.7(a) shows how it works to serve each C_i 's data by enforcing each frame to share the same amount of data $\frac{S_i(t_b)}{T_i}$ for C_i . Clearly, such a scheduling is feasible because the total load in each frame cannot exceed B (otherwise, the BS will be overloaded) and the delay of $Data_i$ is exactly T_i frames.



(a)



(b)

Figure 4.7: Proof of Theorem 4.2.3. (a) the imaginary scheduling and (b) two changes to the imaginary scheduling by our earliest-next-bandwidth-first scheduling.

Now, with our earliest-next-bandwidth-first policy, there are two possible changes: (i) data may be moved to the free space of an earlier frame and (ii) data with a higher priority may squeeze into the space of data with a lower priority. The former has no impact on delay bound. The latter has impact on those with lower priorities. However, if the space of lower-priority data is exchanged with the space of higher-priority data, the former can still make the requested delay bound because the latter is already bounded by a tighter delay bound (this is what “earliest-next-bandwidth” means). Fig. 4.7(b) illustrates the ideas. The arrows on the left-hand side are the movement directions of data toward earlier free space (item (i)). Arrows on the right-hand side are the data exchanges due to their priorities (item (ii)). In this example, high-priority $Data_k$ at frame $t_b + 1$ with $pty(Data_k) = t_b + T_k$ is exchanged with low-priority $Data_i$ at frame t_b with $pty(Data_i) = t_b + T_i$. $Data_i$ can still make its deadline because the space of $Data_k$ already meets a deadline, which is earlier than $Data_i$'s. It follows that $\Delta t \leq T_i$ for all C_i 's. \square

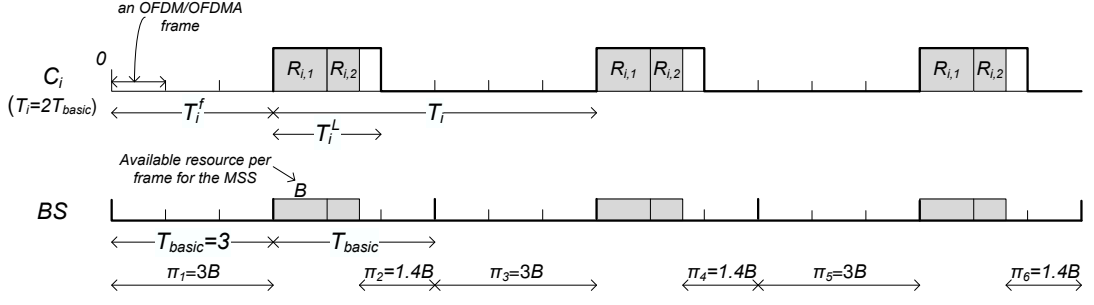


Figure 4.8: Output parameters T_i , T_i^L , T_i^f , and $R_{i,j}$ after the sleep scheduling of C_i and the system parameters π_k , $k = 1.. \frac{T_n}{T_{basic}}$, maintained by the central BS.

4.3 Per-Flow Sleep Scheduling (PSS) Method

In the following, we present our *Per-Flow Sleep Scheduling (PSS)* algorithm. Our goal is to determine the following parameters for each C_i , $i = 1..n$: (1) the cycle length T_i , (2) the listening window T_i^L in each cycle, (3) the starting frame T_i^f of the first cycle, and (4) the amount of resource $R_{i,j}$ to be allocated to the j -th active frame of each cycle, $j = 1..T_i^L$. The calculation should be done at the BS side. Our scheme will maintain a property that the cycle length T_i of each C_i is an integer multiple of the previous T_{i-1} . Therefore, we will sometimes call T_1 the basic cycle T_{basic} . The goal is to increase the overlapping of these listening windows so as to reduce the MS's duty cycle. Assuming that T_{basic} is known, our PSS algorithm involves an iterative process for $i = 1..n$, where each iteration has two steps: (i) determine T_i of each C_i and (ii) schedule $R_{i,j}$, T_i^L , and T_i^f of each C_i . We will discuss how to determine the basic cycle T_{basic} later on. Below, we present some observations, which will serve as guidelines of our design.

Observation 4.3.1. *The resource of bandwidth B bits per frame allocated to an MS can be regarded as an infinite sequence S of a period of one frame. S can be divided into p sub-sequences S_i^p , each with a period of p frames and a resource of B bits per cycle, where $i = 1..p$ and p is a positive integer. Alternatively, S can be divided into m sub-sequences S_i^{p,k_i} , each with a period of p and a resource of $k_i \times B$ bits per cycle, where $i = 1..m$, $m < p$, k_i is a positive integer, and $\sum_{i=1..m} k_i = p$.*

Observation 4.3.2. *A sub-sequence S_i^p can be further divided into p' sub-sequences $S_{i,j}^{p \times p'}$, each with a period of $p \times p'$ frames and each shifted by a distance of p , where $j = 1..p'$. Similarly, a sub-sequence S_i^{p,k_i} can be further divided into p' sub-sequences $S_{i,j}^{p \times p',k_i}$, $j = 1..p'$, with similar properties except that in each cycle the amount of resource is $k_i \times B$ bits.*

Observation 4.3.3. *The scheduling problem for a connection C_i can be regarded as plac-*

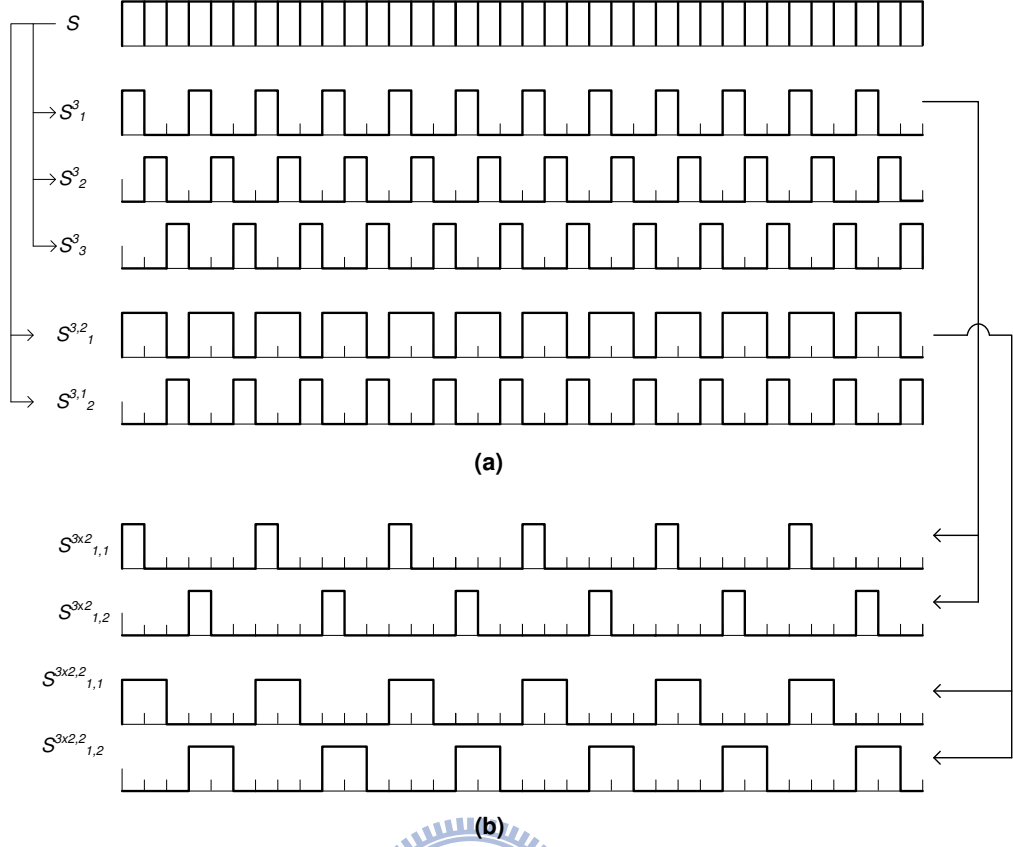


Figure 4.9: Examples of (a) Observation 4.3.1 and (b) Observation 4.3.2.

ing its demand on a sub-sequence with a proper period and resource per cycle. We propose two strategies toward this goal. The first one, called *Delay Bound-based (DB-based) strategy*, tries to accumulate a connection's traffics as much as possible until reaching the delay bound and serve the connection by a sub-sequence with a period slightly tighter than the delay bound and a resource sufficient for the accumulated traffics per cycle. In this way, there are less active frames incurred by the connection. The second one, called *Packet Inter-arrival-based (PI-based) strategy*, tries to serve a connection's traffics immediately once a packet arrives by a sub-sequence with a period slightly tighter than the packet inter-arrival time. If each packet size is small, we may overlap multiple connections' active frames and serve their traffics by one or few active frames.

Observations 4.3.1 and 4.3.2 indicate some ways to decompose the resource allocated to an MS. Fig. 4.9(a) shows two examples. With $p = 3$, S is divided into three subsequences S_1^3 , S_2^3 , and S_3^3 , each with a period of 3 frames. Alternatively, with $m = 2$, S can be divided into two subsequences $S_1^{3,2}$ and $S_2^{3,1}$, which have B and $2B$ bits per cycle, respectively. Following Observation 4.3.2, Fig. 4.9(b) shows how to decompose S_1^3 into $p' = 2$ sub-sequences $S_{1,1}^{3 \times 2}$ and $S_{1,2}^{3 \times 2}$, each with a period of 3×2 , and how to decompose $S_2^{3,2}$ into $S_{2,1}^{3 \times 2,2}$ and $S_{2,2}^{3 \times 2,2}$.

Our PSS follows these observations to pack the traffics of connections together to reduce energy consumption. It has two steps and adds C_i one-by-one to the MS such that the additional active frames for the MS is as few as possible. A data structure π_k , $k = 1.. \frac{T_n}{T_{basic}}$, is maintained to record the amount of remaining free resource in the k -th basic cycle. Initially, $\pi_k = B \times T_{basic}$.

4.3.1 Determining T_i of each C_i

We propose two approaches for this step. The first one, called *PSS-DB (PSS by delay bound)*, sorts C_i s by their packet delay bounds such that $D_1 \leq D_2 \leq \dots \leq D_n$. The second one, called *PSS-PI (PSS by packet inter-arrival time)*, sorts C_i s by their packet inter-arrival times such that $PI_1 \leq PI_2 \leq \dots \leq PI_n$. The design philosophy is in accordance with Observation 4.3.3.

For PSS-DB, we let $T_1 = T_{basic}$ and set T_i for $i = 2..n$ as follows:

$$T_i = T_{i-1} \times \left\lfloor \frac{D_i}{T_{i-1} \times F} \right\rfloor. \quad (4.11)$$

Eq. (4.11) sets T_i as a positive integer multiple of the previous T_{i-1} . In fact, our assignment guarantees in a recursive manner that $T_{i-1} \leq \left\lfloor \frac{D_{i-1}}{F} \right\rfloor$. The initial T_1 would satisfy this condition (to be shown later on). Since $D_{i-1} \leq D_i$, $\left\lfloor \frac{D_i}{T_{i-1} \times F} \right\rfloor$ in Eq. (4.11) must be a positive integer. Also, Eq. (4.11) implies that $T_i \leq T_{i-1} \times \frac{D_i}{T_{i-1} \times F} = \frac{D_i}{F}$. Since T_i is an integer, $T_i \leq \left\lfloor \frac{D_i}{F} \right\rfloor$ meets the delay bound for C_i .

For PSS-PI, we assume that $PI_i \leq D_i$ (this is usually true for most of delay-tolerant real-time applications). We let $T_1 = T_{basic}$ and set T_i for $i = 2..n$ as follows:

$$T_i = \max \left\{ T_{basic}, T_{i-1} \times \left\lfloor \frac{PI_i}{T_{i-1} \times F} \right\rfloor \right\}. \quad (4.12)$$

Eq. (4.12) also sets T_i as a positive integer multiple of the previous T_{i-1} . Our assignment guarantees in a recursive manner that $T_i = T_{basic} \leq \left\lfloor \frac{\min_{i=1..n} \{D_i\}}{F} \right\rfloor$ if $PI_i \leq \min_{i=1..n} \{D_i\}$ and $T_i \leq \left\lfloor \frac{PI_i}{F} \right\rfloor$ otherwise. The initial T_1 would satisfy the former (to be shown later). Since PI_i s are sorted in an ascending order, Eq. (4.12) will force those C_i s such that $PI_i \leq \min_{i=1..n} \{D_i\}$ to choose their $T_i = T_{basic}$. The rest of the C_i s will satisfy the later condition since $T_i \leq T_{i-1} \times \frac{PI_i}{T_{i-1} \times F} = \frac{PI_i}{F}$. It follows that all C_i s will meet their delay bounds because $PI_i \leq D_i$.

Theorem 4.3.1. *In both PSS-DB and PSS-PI, it is guaranteed that each T_i is a positive integer multiple of $T_1 = T_{basic}$, $i = 2..n$, and each C_i 's cycle meets its delay bound, i.e., $T_i \leq \left\lfloor \frac{D_i}{F} \right\rfloor$, $i = 1..n$.*

4.3.2 Scheduling $R_{i,j}$, T_i^L , and T_i^f of each C_i

This step is the same for PSS-DB and PSS-PI. So we will not distinguish between them. Recall the data structure π_k , $k = 1.. \frac{T_n}{T_{basic}}$. We will sequentially schedule C_i , $i = 1..n$, by updating π_k . Specifically, when C_i is under consideration, we will pick one basic cycle among all $\frac{T_n}{T_{basic}}$ basic cycles as the starting point and examine the subsequent basic cycles. For each basic cycle being examined, its remaining resource is allocated; this repeated until we have allocated sufficient resource for C_i . Among all starting points, the one which causes the least increment on the number of active frames is selected. The detail procedure for placing C_i 's demand is as follows:

- a) Calculate the required resource γ_i of C_i per T_i by

$$\gamma_i = \left\lceil \frac{T_i \times F}{PI_i} \right\rceil \times S_i. \quad (4.13)$$

- b) Recall that each T_j is an integer multiple of T_{j-1} , $j = 2..n$. So after placing C_1 's, C_2 's, \dots , and C_{i-1} 's demands, the sequence π_k , $k = 1.. \frac{T_n}{T_{basic}}$, has a period of $\frac{T_{i-1}}{T_{basic}}$. To place C_i 's demand, we only need to check the first $\frac{T_i}{T_{basic}}$ basic cycles as T_i 's starting point. Specifically, for $k = 1$ to $\frac{T_i}{T_{basic}}$ with $\pi_k > 0$, we compute a cost function $f(k)$ to represent the extra active frames incurred to the MS if we place C_i 's demand on the k -th and subsequent basic cycles. Note that since the listening window of a PSC must be continuous, the resources allocated to C_i must be continuous (i.e., we will not leave a frame unallocated if there are frames being allocated before and after the frame).
- c) Let k^* be the index which induces the smallest cost function $f(k)$ in step b. We will place C_i 's demand starting from the k^* -th basic cycle. In case that there is a tie, we will give priority to the one which leaves the least remaining resource in the last frame where C_i 's demand is placed.
- d) Then we set $T_i^f = k^* \times T_{basic} + 1$ and set T_i^L to the number of frames from the first to the last frame where C_i 's demand is placed. Also, $R_{i,j}$ is set accordingly. Finally, we update the remaining free resources in π_k , $k = 1..n$, by subtracting from them the amounts of resources allocated to C_i (note that since the period is T_i , $\pi_k = \pi_{k+\ell \frac{T_i}{T_{basic}}}$, $\ell = 1.. \left(\frac{T_n}{T_i} - 1 \right)$).

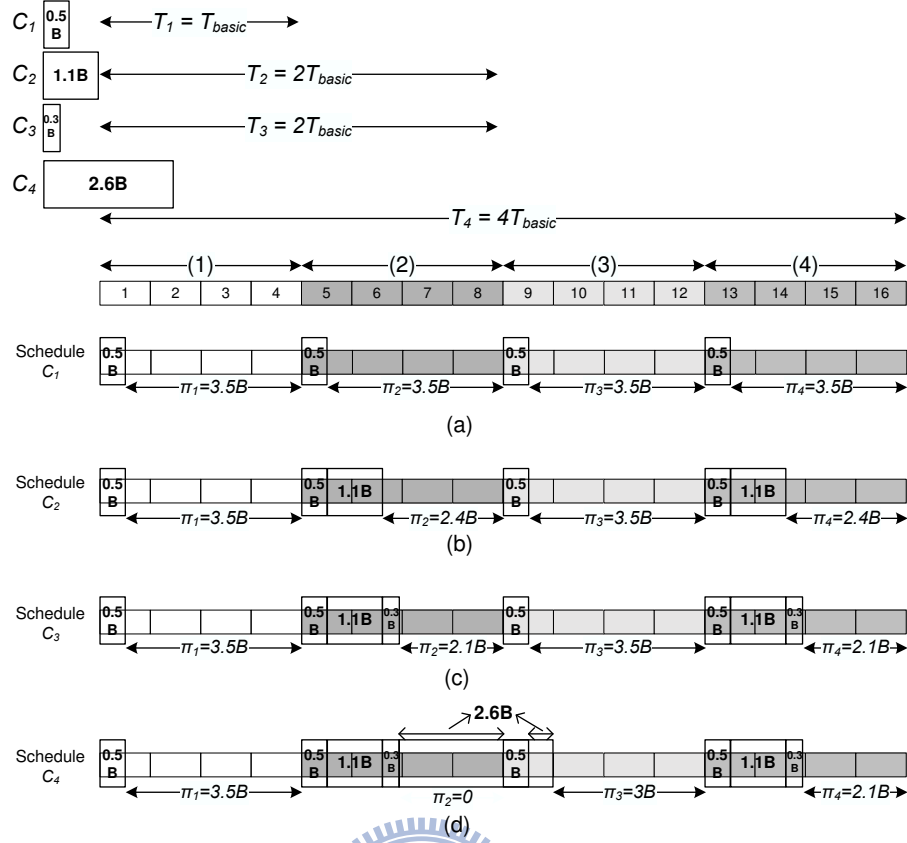


Figure 4.10: Example of scheduling $R_{i,j}$, T_i^L , and T_i^f of four connections in the MS.

Example 4.3.1. Fig. 4.10 shows an example of step 2. The MS contains 4 connections C_1 , C_2 , C_3 , and C_4 with sleeping cycles of $T_1 = T_{basic}$, $T_2 = 2T_{basic}$, $T_3 = 2T_{basic}$, and $T_4 = 4T_{basic}$ and required resources per cycle of $\gamma_1 = 0.5B$, $\gamma_2 = 1.1B$, $\gamma_3 = 0.3B$, and $\gamma_4 = 2.6B$, respectively, where $T_{basic} = 4$ frames. Initially, $\pi_k = 4B$ for $k = 1..4$. Then, each C_i is scheduled as follows. For C_1 , any selection of k^* is the same for it. So we set $k^* = 1$, $R_{1,1} = 0.5B$, $T_1^f = 1$, and $T_1^L = 1$. The BS reserves $\gamma_1 = 0.5B$ resource for C_1 in every basic cycle as shown Fig. 4.10(a), so $\pi_1 = \pi_2 = \pi_3 = \pi_4 = 3.5B$. For C_2 , its k^* can be 1 or 2. Allocating γ_2 in the first or second basic cycle would add one more active frame to the MS (i.e., $f(1) = f(2) = 1$) and leave the same remaining resource of $0.4B$ in the last frame. So we randomly select $k^* = 2$, which gives $R_{2,1} = 0.5B$, $R_{2,2} = 0.6B$, $T_2^f = 5$, and $T_2^L = 2$, as shown in Fig. 4.10(b). Then we update $\pi_1 = 3.5B$, $\pi_2 = 2.4B$, $\pi_3 = 3.5B$, $\pi_4 = 2.4B$. For C_3 , choosing $k^* = 1$ or 2 would require no additional active frame for the MS (i.e., $f(1) = f(2) = 0$). But setting $k^* = 2$ would leave less remaining resource in the last frame (i.e., $0.1B$). So we set $k^* = 2$ and update $R_{3,1} = 0.3B$, $T_3^f = 6$, and $T_3^L = 1$, as shown in Fig. 4.10(c). Then we update $\pi_1 = 3.5B$, $\pi_2 = 2.1B$, $\pi_3 = 3.5B$, $\pi_4 = 2.1B$. For C_4 , setting $k^* = 2$ or 4 would add less active frames (i.e., $f(2) = f(4) = 2 < f(1) = f(3) = 3$). Since $k^* = 2$ and 4 will both leave the same remaining resource in the last frame, we randomly pick $k^* = 2$. So we set $R_{4,1} = 0.1B$,

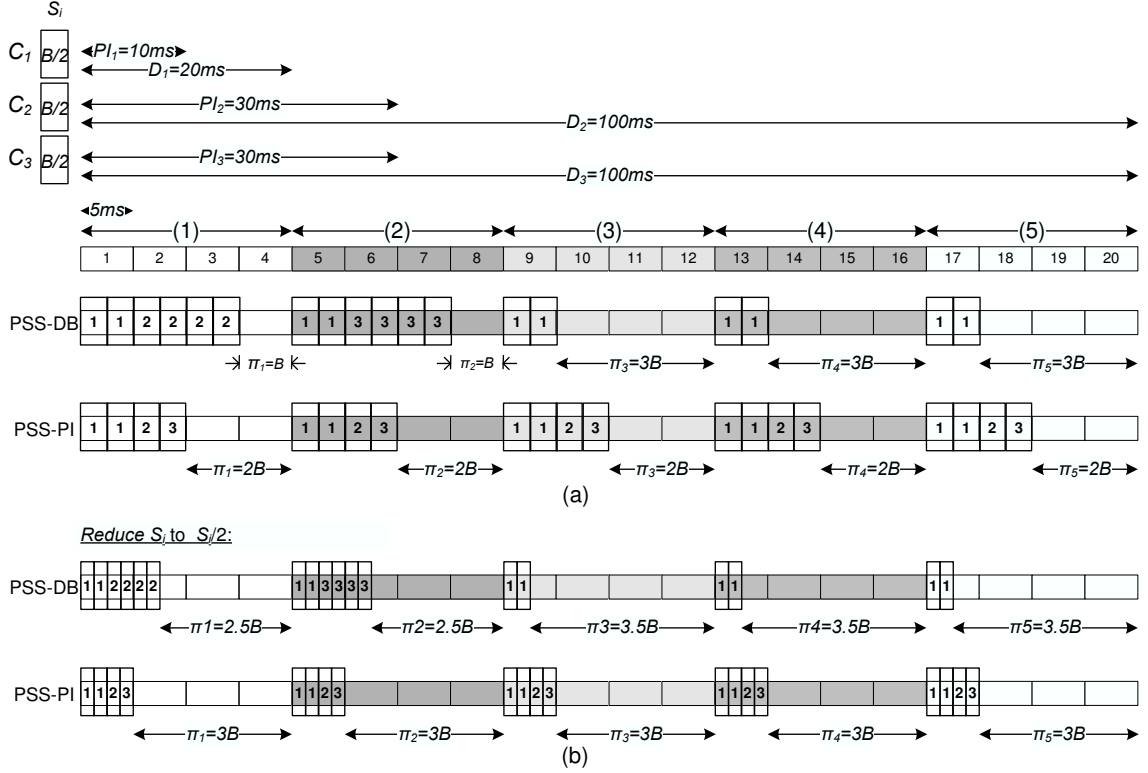


Figure 4.11: Comparisons of PSS-DB and PSS-PI under different S_i (a) S_i , (b) $\frac{S_i}{2}$.

$R_{4,2} = 1B$, $R_{4,3} = 1B$, $R_{4,4} = 0.5B$, $T_4^f = 6$, and $T_4^L = 4$, as shown in Fig. 4.10(d). Then we update $\pi_1 = 3.5B$, $\pi_2 = 0$, $\pi_3 = 3B$, and $\pi_4 = 2.1B$.

Example 4.3.2. Fig. 4.11 uses an example to compare PSS-DB and PSS-PI. In Fig. 4.11(a), the MS contains 3 connections C_1 , C_2 , and C_3 with packet inter-arrival times of $PI_1 = 10\text{ms}$, $PI_2 = 30\text{ms}$, and $PI_3 = 30\text{ms}$, delay bounds of $D_1 = 20\text{ms}$, $D_2 = 100\text{ms}$, and $D_3 = 100\text{ms}$, and packet sizes of $S_1 = \frac{B}{2}$, $S_2 = \frac{B}{2}$, and $S_3 = \frac{B}{2}$. Fig. 4.11(b) changes the packet sizes to $S_1 = \frac{B}{4}$, $S_2 = \frac{B}{4}$, and $S_3 = \frac{B}{4}$. Fig. 4.11(a) shows that PSS-DB will consume 9 active frames per 20 frames and PSS-PI will consume 10 active frames per 20 frames, while Fig. 4.11(b) shows that PSS-DB will consume 7 active frames per 20 frames and PSS-PI will consume only 5 active frames per 20 frames. Intuitively, PSS-DB can pack packets of a connection together according to their delay bound and is more favorable when S_i s are relatively closer to B . Contrarily, PSS-PI tries to pack packets of different connections together and serve them immediately after their arrival and is more favorable when S_i s are relatively smaller than B .

Lastly, we present how to determine the basic cycle T_{basic} . Since $T_{basic} = T_1$ and T_1 must satisfy $T_1 \leq \lfloor \frac{D_1}{F} \rfloor$ for PSS-DB and $T_1 \leq \left\lfloor \frac{\min_{i=1..n} D_i}{F} \right\rfloor$ for PSS-PI, we propose to pick T_{basic} from the interval $[1, \lfloor \frac{D_1}{F} \rfloor]$ for PSS-DB and the interval $[1, \left\lfloor \frac{\min_{i=1..n} D_i}{F} \right\rfloor]$ for PSS-

PI. For each candidate T_{basic} , we adopt an exhausted search to compute a cost function $g(T_{basic})$ to represent the ratio of active frames of the MS if T_{basic} is used. Let T_{basic}^* be the basic cycle which induces the least cost. Then this T_{basic}^* is chosen for T_1 .

4.4 Feasibility Study of the Scheduling Problem

The above discussion did not answer the question: “What happens when a feasible scheduling can not be found?” Below, we conduct some feasibility study of the sleep scheduling problem. It is not hard to see that given a set of connections, if their traffics can be satisfactorily arranged without violating their deadlines, then “always active” is a straightforward schedule (note that this statement does not imply whether the scheduling is energy-efficient or not). Below, we show that deciding whether traffics of a set of connections can be satisfactorily scheduled can be reduced to a maximum matching problem, which is computationally tractable. Solutions for maximum matching can be found in [12]. Thus, deciding whether a scheduling problem is feasible has a polynomial-time solution (following the above note, it remains a question whether the solution is most energy-efficient).

We are still given C_i and its PI_i , D_i , and S_i , $i = 1..n$. In our derivation, we assume that the units of PI_i and D_i are frames and the units of S_i and the resource B are bits. Let $L = lcm\{PI_i, i = 1..n\}$. Note that traffic arrivals repeat at the period of L frames. Below, we will model our scheduling problem as a maximum matching problem in a bipartite graph $G = (\{V_l, V_r\}, E)$ as defined below.

1. For each C_i , consider its packet arrivals during L frames. There are $\frac{L}{PI_i} \times S_i$ bits. So we construct for C_i the same number of vertices, denoted by $C_{i,j,k}$, $j = 1..\frac{L}{PI_i}$ and $k = 1..S_i$, in V_l . So $|V_l| = \sum_{i=1}^n \left(S_i \times \frac{L}{PI_i} \right)$.
2. For the B bits in continuous L frames, we construct the same number of vertices, denoted by $F_{x,y}$, $x = 1..L$ and $y = 1..B$, in V_r . So $|V_r| = B \times L$.
3. We regard vertex $C_{i,j,k} \in V_l$ as the k -th bit of the j -th packet of connection C_i . Let $t_{i,j}$ be the frame index of its arrival. Then it has to be delivered by frame $t_{i,j} + D_i - 1$. So we construct edges $(C_{i,j,k}, F_{x,y})$ in E for $x = t_{i,j} + 1..t_{i,j} + D_i - 1 \pmod{L}$ and $y = 1..B$. Each edge means that $C_{i,j,k}$ can be assigned to resource $F_{x,y}$ without violating the delay bound. Note that some x may exceed L , so we use “ \pmod{L} ” to represent the subsequent L frames in the next round, i.e., they are represented by wrap-around edges.

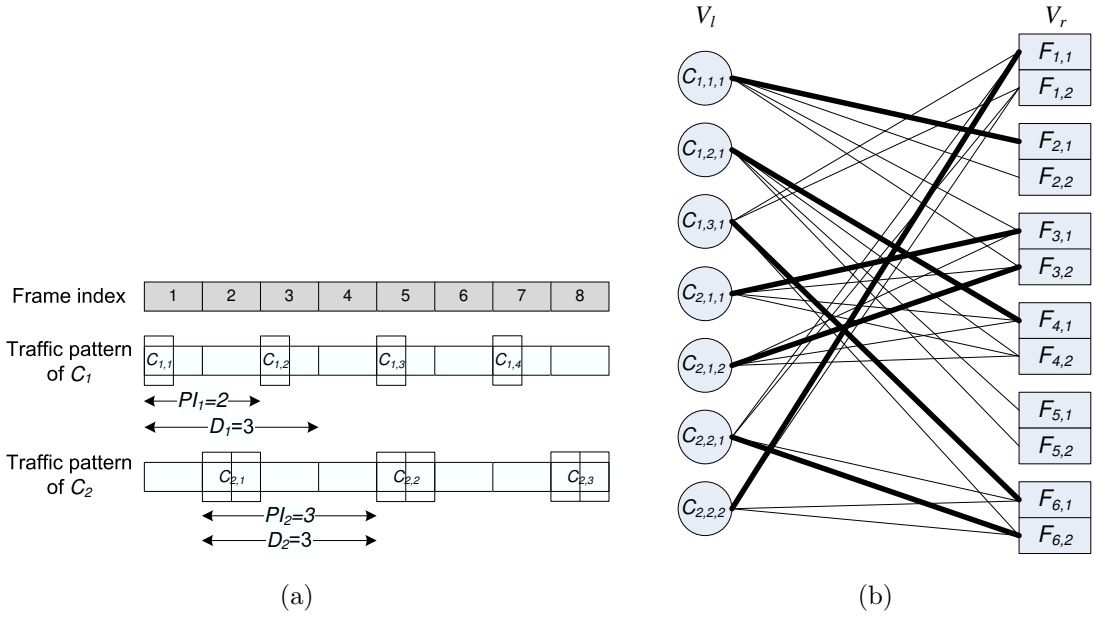


Figure 4.12: Example of modeling a scheduling problem as a maximum matching problem.

Example 4.4.1. Fig. 4.12(a) shows an example. The MS contains 2 connections C_1 and C_2 with $PI_1 = 2$, $PI_2 = 3$, $D_1 = 3$, $D_2 = 3$, $S_1 = 1$, and $S_2 = 2$. Assuming $B = 2$, we can form a bipartite graph as shown in Fig. 4.12(b). Since $\text{lcm}\{PI_1, PI_2\} = 6$, we consider $\frac{6}{PI_1} = 3$ and $\frac{6}{PI_2} = 2$ packet arrivals of C_1 and C_2 , respectively. Since each packet of C_2 has 2 bits, there are 7 vertices in V_l . In V_r , there are $B \times 6 = 12$ vertices. For example, bits $C_{2,2,1}$ and $C_{2,2,2}$ can be assigned to the sixth frame (of the current round) and the first frame (of the next round). The bold lines in Fig. 4.12(b) show one maximum-matching solution.

Theorem 4.4.1. A scheduling problem has a feasible schedule if and only if its corresponding bipartite graph $G = (\{V_l, V_r\}, E)$ has a maximum matching with size of $|V_l|$.

4.5 Experimental Results

4.5.1 Performance Evaluation of FD Method

We have developed a simulator in C to evaluate the performance of proposed Fold-and-Demultiplex scheme. In our simulation, the frame length $F = 5$ ms and the resource B that can be allocated per frame is a fixed amount in each simulation round. Between the BS and the MS, we define six real-time flow types with different QoS parameters as shown in Table 4.1. The directions of flows of types III~VI are randomly decided as down or up. Types I and II have the same QoS parameters and differ in their directions. For non-real-time flows, we assume their packet arrival following the poisson distribution.

Table 4.1: Six types of real-time flows used in the simulation.

Flow type	Service type	Direction	Delay constraint (ms)	Packet inter-arrival time (ms)	Packet size (byte)
I	ERT-VR	DOWN	50	20	160
II	ERT-VR	UP	50	20	160
III	UGS	DOWN/UP	60	20	160
IV	UGS	DOWN/UP	300	80	500
V	RT-VR	DOWN/UP	300	35	440
VI	RT-VR	DOWN/UP	800	200	2000

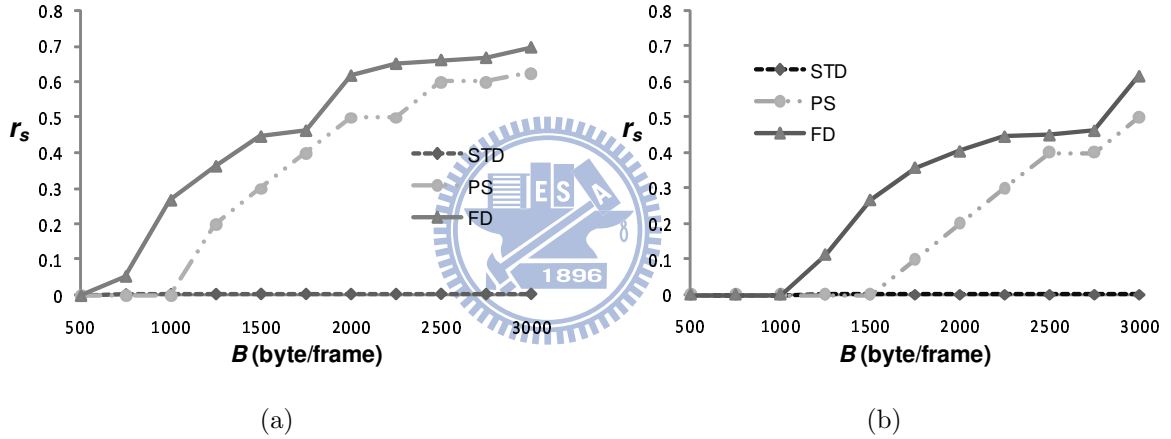


Figure 4.13: r_s vs. B with (a) two real-time flow sets and (b) three real-time flow sets.

We compare our (FD) scheme against the scheme in the standard [26] and the scheme in [61] (denoted as STD and PS, respectively). Note that PS uses one single type II PSC to serve all real-time flows and we enforce the wake-up period of the PSC to be smaller than or equal to the minimum delay constraint of all flows. As to performance metrics, we use *sleep ratio* (r_s), *resource utilization* (U , i.e., how well the bandwidth B is utilized), average delay, and jitter to make comparisons. For non-real-time traffic, only sleep ratio and response time are considered. Below, we define one *real-time flow set* as six flows containing one from each of types I~VI.

Impact of B and Traffic Load

Fig.4.13(a) and Fig.4.13(b) plot r_s against B for STD, PS, and FD schemes with two and three real-time flow sets, respectively. As B increases, r_s increases for both PS and FD.

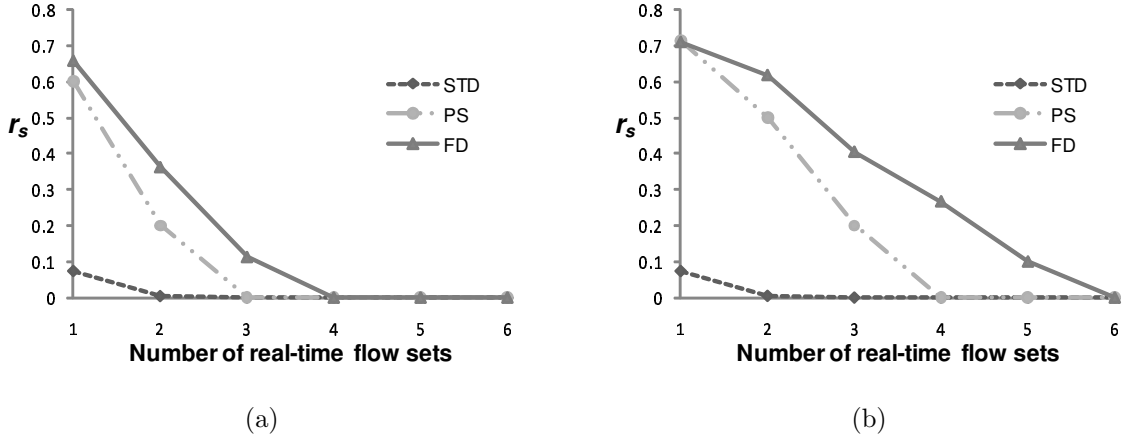


Figure 4.14: r_s vs. number of real-time flow sets with (a) $B = 1250$ byte/frame and (b) $B = 2000$ byte/frame.

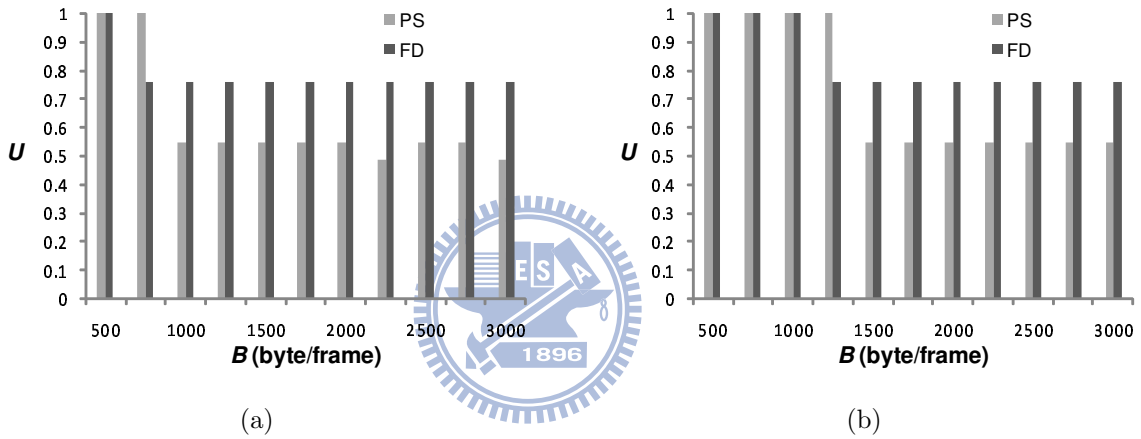


Figure 4.15: U vs. B with (a) two real-time flow sets and (b) three real-time flow sets.

FD always performs the best, followed by PS. This is because FD uses multiple PSCs to manage the sleeping behavior of the MS while PS uses one PSC. This is particularly important when flows have different QoS characteristics. Since PS uses only one PSC, it has to reserve the maximum required resource in each cycle, thus wasting lots of resources sometimes. STD uses too many PSCs, which are not synchronized, leading to very low r_s . By varying the number of real-time flow sets and fixing $B = 1250$ (resp., $B = 2000$) byte/frame, Fig.4.14(a) (resp., Fig.4.14(b)) shows the obtained r_s . Naturally, r_s decreases as the number of real-time flow sets increases. FD can sustain up to 3 (resp., 5) sets when $B = 1250$ (resp., $B = 2000$). On the contrary, with 3 (resp., 4) sets, PS will keep the MS always awake when $B = 1250$ (resp., $B = 2000$). This is because PS over-estimates the potential traffic in some frames. Such an effect is even more serious when there exist larger differences among flows' delay constraints and inter-arrival times.

Fig.4.15(a) and Fig.4.15(b) illustrate the resource utilization U by varying B when

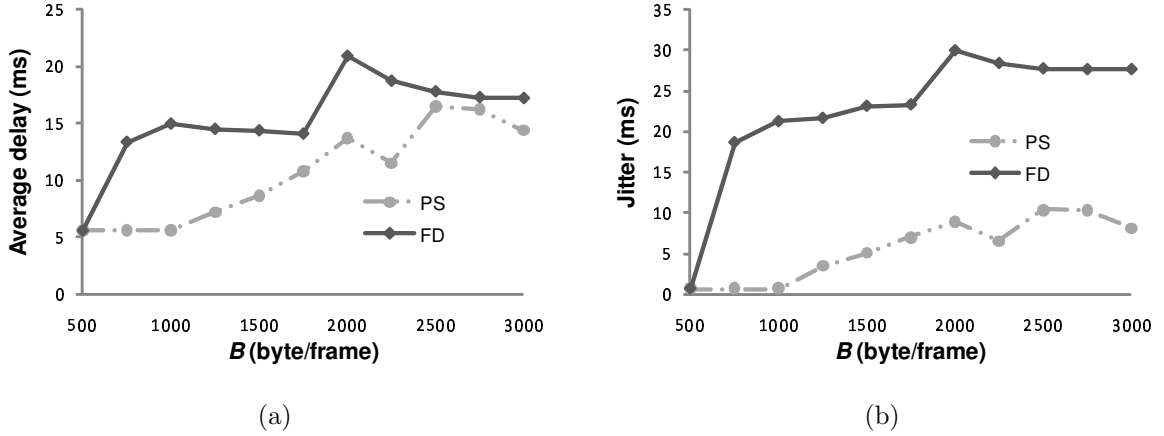


Figure 4.16: (a) Average delay vs. B and (b) jitter vs. B (two real-time flow sets).

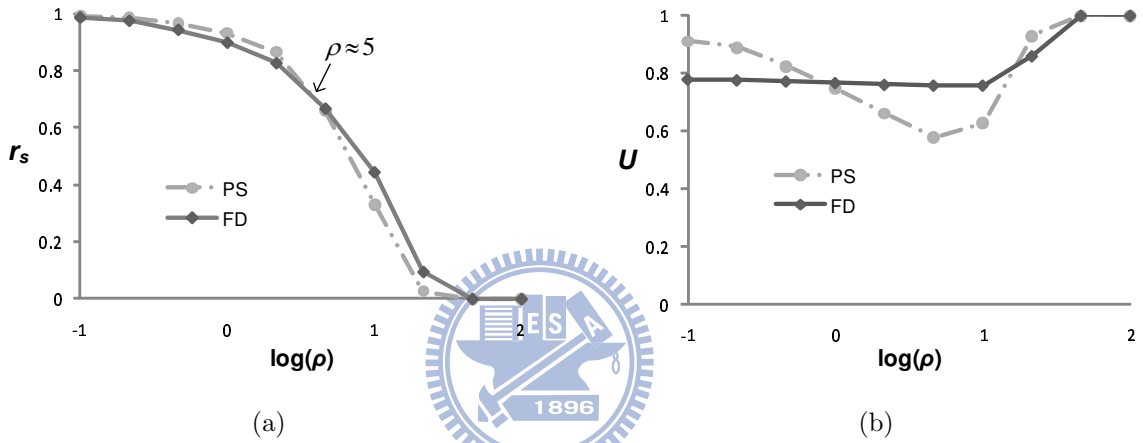


Figure 4.17: Effect of ρ on r_s and U under random flow arrival ($B = 1250$ byte/frame).

there are 2 and 3 real-time flow sets, respectively. We see that FD always outperforms PS, due to PS's over-estimation on resource demands. Fig.4.16(a) and Fig.4.16(b) plot average delay and jitter against B , respectively, where jitter is defined as the standard deviation of packet delivery delay. We see that FD causes higher delay and jitter in all cases because it will buffer packets with larger delay constraints for future delivery. However, as has been clear from our claims, their delay constraints are still guaranteed.

Impact of Flow Arrival Pattern

The above discussion considers flow types of fixed combination. Below, we consider real-time flows of each type arriving at a Poisson process with rate λ_r and exponentially distributed hold time with mean $1/\mu$. Letting $\rho = \frac{\mu}{6\lambda_r}$, we will observe its impact on performance. With $B = 1250$, Fig.4.17(a) shows the impact of ρ on r_s . When ρ is small, PS performs slightly better than FD. After $\rho \geq 5$, FD outperforms PS because a larger ρ causes multiple flows coexisting to show FD's advantage. On the other hand, when ρ is

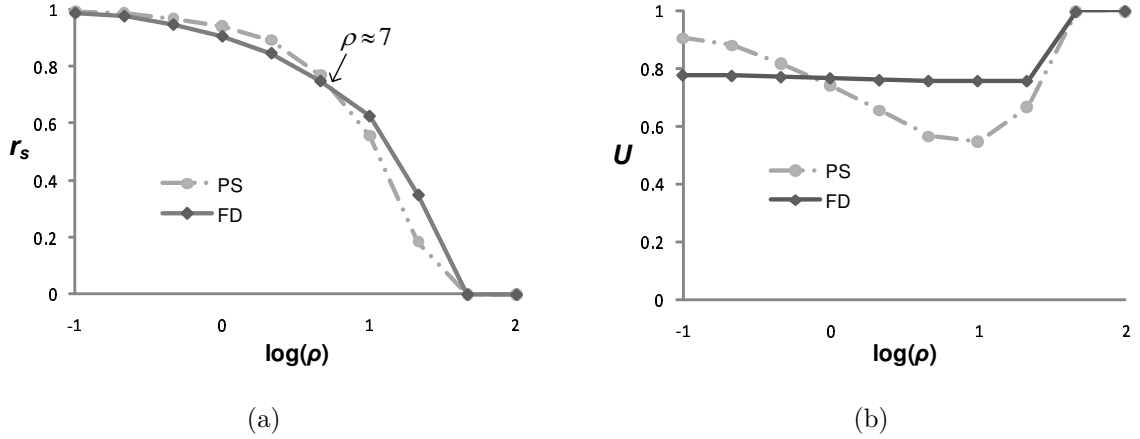


Figure 4.18: Effect of ρ on r_s and U under random flow arrival ($B = 2000$ byte/frame).

small, there is usually 1 or very few flows between the MS and the BS. Therefore, only a single PSC is enough for the sleep operation. Compared to PS's wake-up period, which is close to or even equal to the most strict delay constraint of flows, FD's wake-up period is smaller than or equal to half of the flow's delay constraint. This makes FD awake longer than PS when ρ is small. Fig.4.17(b) plots U against ρ . When $\rho < 1$, PS performs better than FD; after $\rho > 1$, FD becomes better; but after $\rho \geq 20$, PS outperforms FD again. Under $\rho < 1$, there is less than 1 flow in average; since PS uses the delay constraint of the flow as the sleep cycle, it can get better utilization. As there are more flows, FD performs better than PS due to its multi-PSC capability. However, when there are too many flows ($\rho \geq 20$), PS outperforms FD again because PS would disable the sleep mode more often than FD does (here we assume that the resource utilization is 1 when the sleep mode is disabled because resource can be accurately allocated). Fig.4.18 sets $B = 2000$ and shows similar results to Fig.4.17. In Fig.4.18(a), we can see that a larger B makes FD outperform PS after $\rho \geq 7$ which is larger than the value of 5 in the case of $B = 1250$ (Fig.4.17(a)). This is because a larger B has a higher tolerance on PS's over-estimation on resource demand.

Sleep Performance by Including Non-real-time Traffic

In this part, we consider one real-time flow set with a non-real-time downlink traffic flow with packet arrival rate λ_n . To verify the effectiveness of our scheme in handling non-real-time traffic, we simulate FD with and without including a PSC of type I. Fig.4.19 shows r_s and the average response time experienced by non-real-time packets. A larger λ_n will degrade r_s . When $T_{S,max}$ is smaller, adding a PSC of type I can significantly save energy because it is easier to reach the maximum sleep window, after which the type I PSC can reuse the active frames of type II PSCs. In terms of response time, using a type

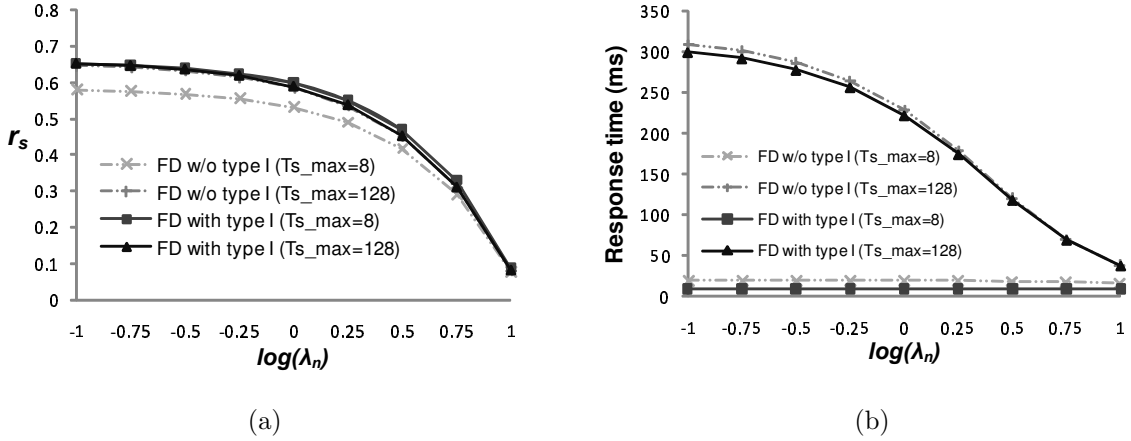


Figure 4.19: Effect of λ_n on (a) r_s and (b) response time with one real-time flow set and a non-real-time downlink flow of rate λ_n ($B = 1250$ byte/frame).

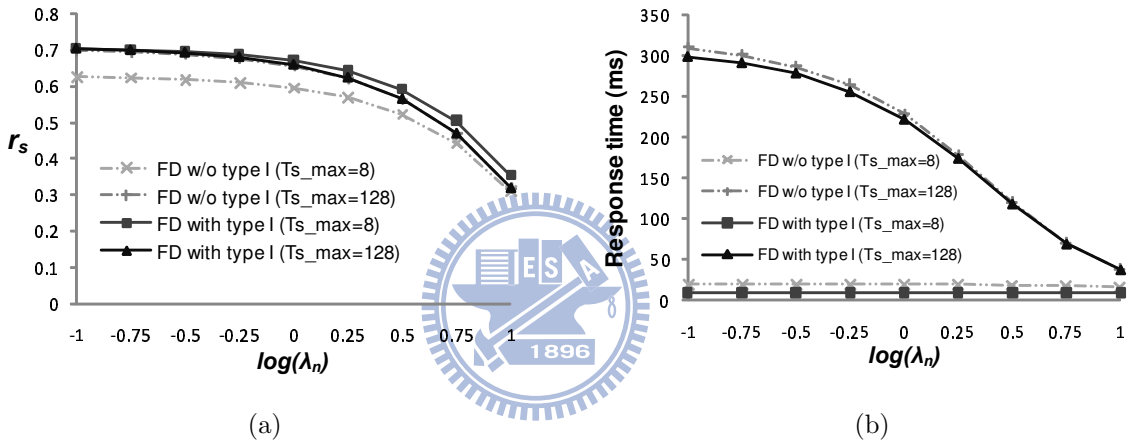


Figure 4.20: Effect of λ_n on (a) r_s and (b) response time with one real-time flow set and a non-real-time downlink flow of rate λ_n ($B = 2000$ byte/frame).

I PSC is always beneficial because FD actually uses a smaller maximum sleep window size than the original given one for the type I PSC. Fig.4.20 sets $B = 2000$ and shows similar results to Fig.4.19.

4.5.2 Performance Evaluation of PSS Method

To verify our result, we have simulated a BS-MS pair with multiple real-time connections. Table 4.2 lists six types of real-time connections used in our simulation. The length of an OFDM/OFDMA frame is set to 5 ms[30]. We consider three performance metrics: (i) *power consumption*: the ratio of active frames for the MS, (ii) *resource utilization*: the ratio of the amount of resource consumed by the MS to the total amount allocated to it, and (iii) *drop rate*: the ratio of dropped packets due to missing deadline. We will compare our PSS-DB and PSS-PI against the PS scheme in [61] and an ideal *power consumption*

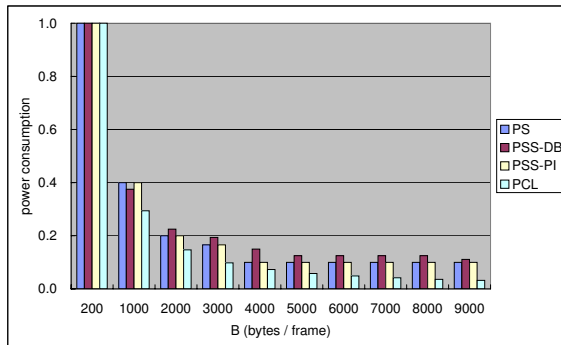
Table 4.2: QoS parameters of different real-time connections.

	Packet size (bytes)	Packet inter-arrival time (ms)	Delay bound (ms)
Type I	24	30	50
Type II	160	20	50
Type III	420	33.33	200
Type IV	1250	33.33	300
Type V	416	30	50
Type VI	1250	[20,200]	[50,500]

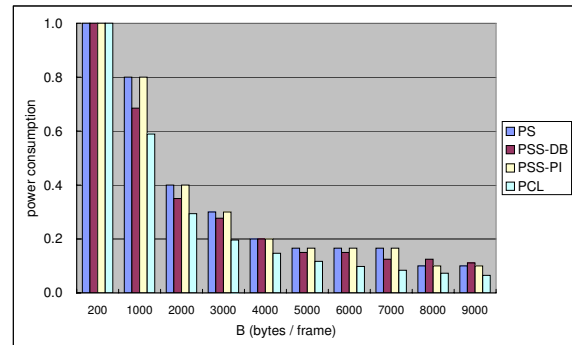
lower bound $PCL = \sum_{i=1}^n \frac{S_i}{PI_i \times B} \times F$, where PCL stands for the lowest active ratio for the MS to support all given connections' traffics. We derive PCL by relaxing the delay bounds of connections as ∞ . Then, we can set the sleep cycle of the MS as $T_c \times \frac{lcm\{PI_i|i=1..n\}}{F}$ frames, where T_c is the minimum integer that makes (1) the sleep cycle be an integer and (2) the arrival data during the sleep cycle fill the frame up (i.e., $\frac{\sum_{i=1}^n \frac{T_c \times lcm\{PI_i|i=1..n\}}{PI_i} \times S_i}{B}$ is a positive integer). Therefore, PCL can be derived by $\frac{\sum_{i=1}^n \frac{T_c \times lcm\{PI_i|i=1..n\}}{PI_i} \times S_i}{T_c \times lcm\{PI_i|i=1..n\}/F}$. Note that PCL provides only the value of power consumption, so we don't compare PSS-DB and PSS-PI against PCL in resource utilization and drop rate.

Effects of B

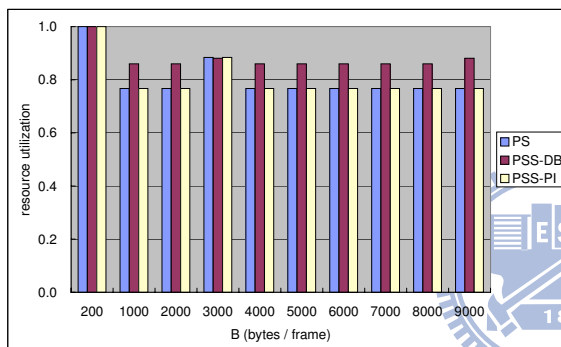
We consider two environments as follows. Environment 1 has four connections containing each of traffic types of I, II, III, and IV. Environment 2 has eight connections, two for each of types I-IV. Fig. 4.21 shows the effect of B on the power consumption, utilization, and drop rate under environments 1 and 2. Generally, as shown in Fig. 4.21(a) and (d), power consumption decreases as B increases. As B increases, we can see that, in the initial, PSS-DB performs the best in all schemes except PCL and has the smallest difference to PCL because PSS-DB can more accurately capture the required resource than PSS-PI and PS schemes such that PSS-DB can consume less active frames; but as B becomes larger, PSS-DB consumes more active frames than PSS-PI and PS because PSS-PI and PS serve packets immediately after their arrival and this is more favorable when packet size and resource requirement are relatively smaller than B . PSS-PI and PS perform the same power consumption because the PIs of traffic types I~IV are close and smaller than the strictest delay bound, causing PSS-PI conducting the same sleep schedule as PS. Comparing Fig. 4.21(a) and (d), we can see when the number of connections is doubled, a larger size of B is needed for PSS-PI and PS to win PSS-DB in power consumption. Note that when $B \leq 250$ (resp., $B \leq 500$) bytes/frame in Fig. 4.21(a) (resp., Fig. 4.21(d)),



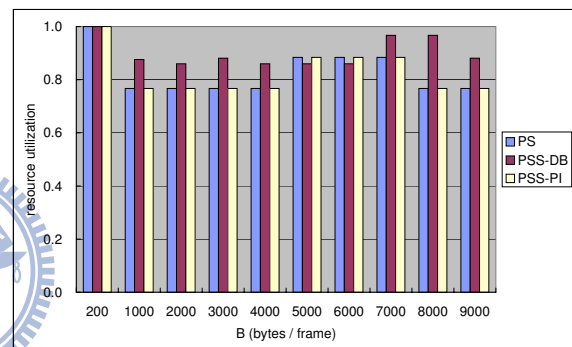
(a)



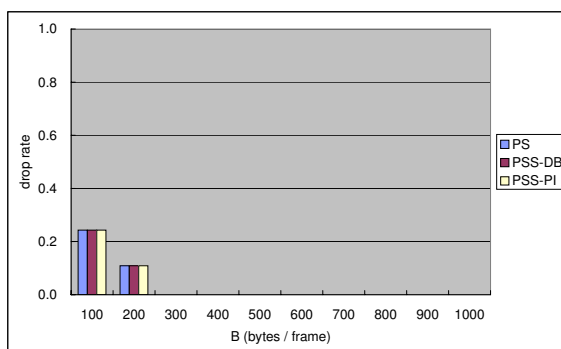
(d)



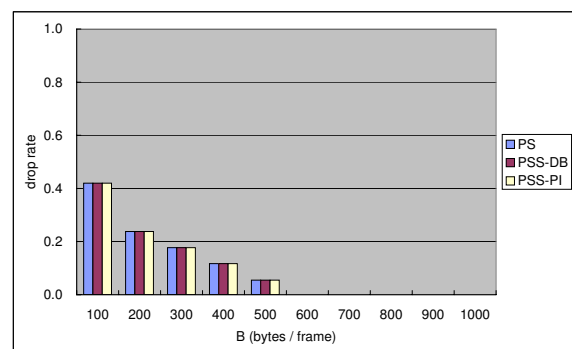
(b)



(e)



(c)



(f)

Figure 4.21: Effects of B on the power consumption, resource utilization, and drop rate under environments 1 ((a)~(c)) and 2 ((d)~(f)).

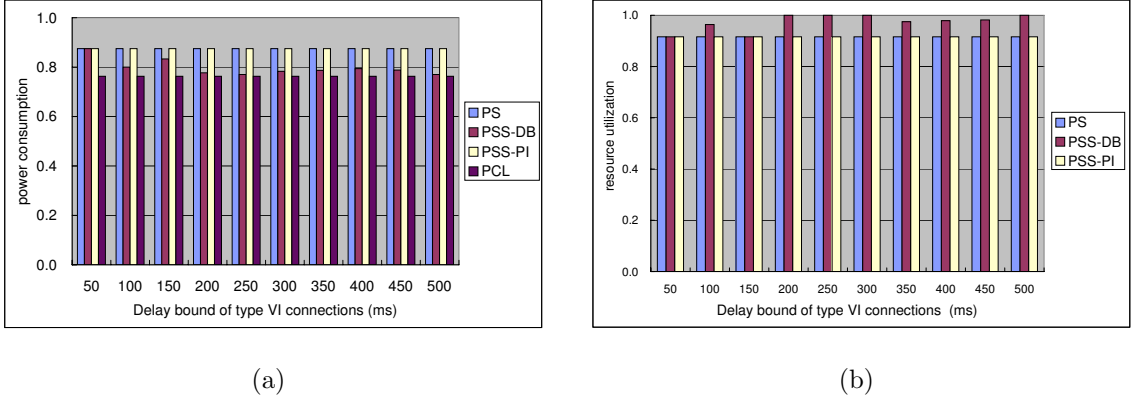


Figure 4.22: Effects of connection delay bound on (a) power consumption and (b) resource utilization with packet inter-arrival time of type VI connections being 20 ms.

resource is not enough for the MS to activate sleep mode; so the power consumption is 100%. Fig. 4.21(b) and (e) show the resource utilization over different B . PSS-DB is always more than 86% while PSS-PI and PS perform unstable (76% ~ 88%). This shows that PSS-DB can more precisely capture the resource requirement of connections than PSS-PI and PS. Fig. 4.21(c) and (f) show the drop rate of each scheme, we can see that there are no difference for the three schemes, PSS-DB, PSS-PI, and PS, in the drop rate because packets are dropped when the resource is not enough for the MS ($B \leq 250$ bytes/frame for environment 1 and $B \leq 500$ bytes/frame for environment 2). So, our proposed schemes can guarantee the delay bound of packets like PS.

Effects of Connection Delay Bound

We then investigate the effect of connection delay bound on the power consumption and utilization by fixing $B = 1000$ bytes/frame. There are four connections in the MS, two for each of types V and VI. Then we evaluate the performance of the three schemes by increasing the delay bound of type VI connections from 50 ms to 500 ms and fixing their packet inter-arrival time as 20 ms. In Fig. 4.22(a), PSS-DB shows the best power consumption than PSS-PI and PS and has the smallest difference to PCL. After the delay bound over 200 ms, the power consumption of PSS-DB is very close to that of PCL. This means PSS-DB almost reaches the optimal power consumption. PSS-PI and PS show the same performance and no improvement as the delay bound increases because performance of PS is bounded by the strictest delay bound of connections and PSS-PI is bounded by both the strictest delay bound and packet inter-arrival times of connections. Fig. 4.22(b) shows that PSS-DB performs the best utilization than PSS-PI and PS because it evaluates required resource of connections by delay bound, causing least resource waste.

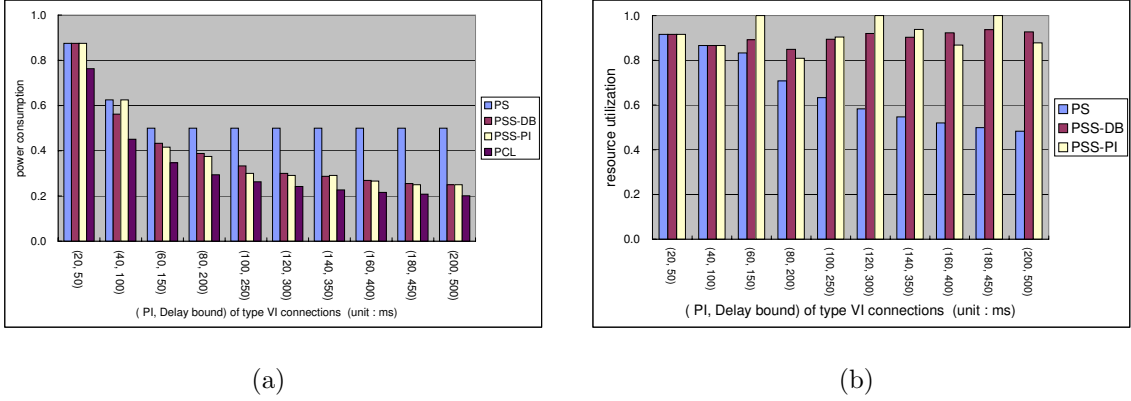


Figure 4.23: Effects of connection packet inter-arrival time and delay bound on (a) power consumption and (b) resource utilization.

Effects of Connection Packet Inter-arrival Time and Delay Bound

In this experiment, we investigate the effect of connection packet inter-arrival time and delay bound on the power consumption and utilization by fixing $B = 1000$ bytes/frame. There are four connections in the MS, two for each of type V and VI. Then we increase both the packet inter-arrival time and the delay bound of type VI connections from 20 ms to 200 ms and 50 ms to 500 ms, respectively. Usually, the delay bound of a voice connection is 2.5 ~ 3 times the size of its packet inter-arrival time while the delay bounds of a multimedia connection and other types of connections are more than 3 times the size of the packet inter-arrival time. In this experiment, we set the delay bound of type VI connections as 2.5 times the size of the packet inter-arrival time. Fig. 4.23(a) shows the power consumption decreases when the packet inter-arrival time and delay bound increase. Both PSS-DB and PSS-PI perform better than PS because our approaches assign each connection a PSC such that the resource requirement can be more accurately captured. As the variances of packet inter-arrival times and delay bounds between connections become larger, i.e., packet inter-arrival time and delay bound of type VI connection become larger, PSS-DB and PSS-PI consume less and less active frames than PS. The power consumption of PSS-DB and PSS-PI even improve 50% when packet inter-arrival time and delay bound of type VI connections are 200 ms and 500 ms, respectively. In Fig. 4.23(b), we can see that PSS-DB and PSS-PI always keep high utilization. However, the utilization of PS decreases when the packet inter-arrival time and delay bound of type VI connections increase because single PSC can not capture the traffic characteristics of connections, especially in the case that the variances of packet inter-arrival times and delay bounds between connections are large. This shows that PSS-DB and PSS-PI can capture the connections' traffic characteristics and waste much less allocated resource than PS.

Chapter 5

Power Saving Class Management for Multiple MSs in Mobile Broadband Networks

In this chapter, focusing on multiple MSs under a BS, we propose to assign PSCs to MSs according to their QoS characteristics so that the total active frames of a MAN (metropolitan area network) can be minimized. In the related work, reference [24] has proposed to form single type II PSC for each MS where the PSCs all use the same sleep cycles. That is, it only considers the strictest delay bound of MSs. Simulation results show that our approach can have much less power consumption than the previous schemes.

5.1 Motivation and Problem Definition

In this section, we introduce the motivations and then present the problem definition. For PSCs of type II and multiple MS sleep scheduling, previous work [24] assigns each MS a PSC with the same common sleep cycle, which is the strictest delay bound of the MSs, and schedules their resource allocation in an overlap free manner. For the system, this kind of solutions is simple and easy to implement. However, if we can assign each MS a sleep cycle which matches its delay bound, it has a great possibility that the overall power consumption of the system and the per-MS active time can be further reduced. Fig. 5.1 shows an example with two MSs M_1 and M_2 , which have data arrival rate of $\tau_1 = 0.2\Omega/\text{frame}$ and $\tau_2 = 0.075\Omega/\text{frame}$ and delay bounds of $D_1 = 4$ (frames) and $D_2 = 12$ (frames), respectively, where Ω is the capacity of a single OFDM/OFDMA frame. As shown in Fig. 5.1(a), by using the sleep scheduling method in [24], a common sleep cycle T_{com} of $\min(D_1, D_2) = 4$ frames is used. Then, M_1 and M_2 both have to wake up for 1 frame per four frames. Reserved resource per T_{com} for M_1 and M_2 is $\tau_1 \times T_{com} = 0.8\Omega$ and $\tau_2 \times T_{com} = 0.3\Omega$, respectively. As Fig. 5.1(b) shows, by assigning

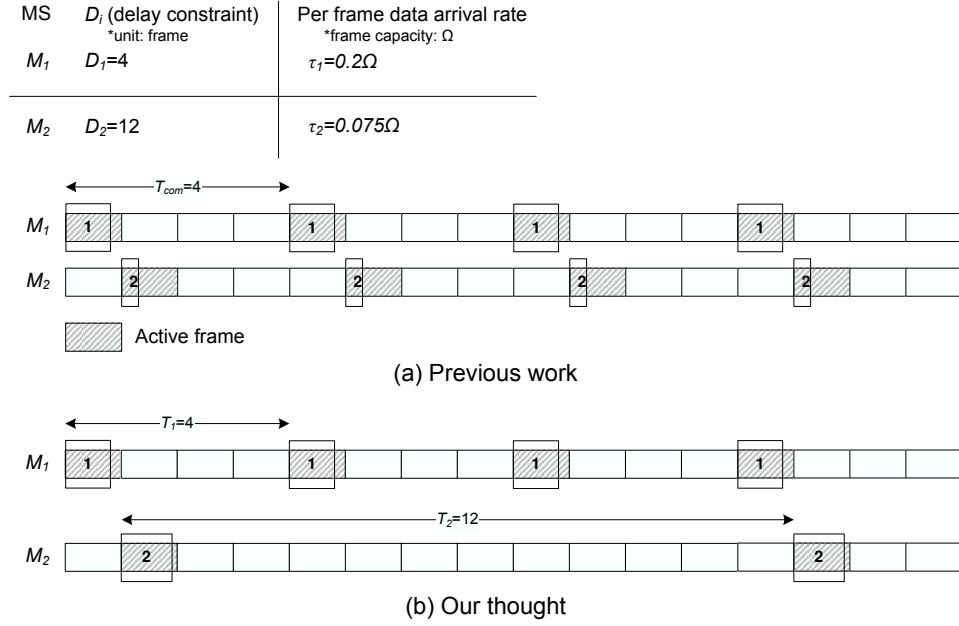


Figure 5.1: Sleep scheduling for two MSs MS_1 and MS_2 using (a) the same common sleep cycle and (b) each with its own sleep cycle.

each MS a sleep cycle adapting to its delay bound, we can schedule the sleep cycle of M_1 and M_2 as 4 and 12 frames, respectively. M_1 and M_2 need 0.8Ω and 0.9Ω of resource per cycle, respectively. By scheduling M_1 and M_2 being awake in different frames, the overall active frames of the system is the least and for both M_1 and M_2 , they will both need 1 active frame per cycle. So, Fig. 5.1(a) needs 6 active frames per 12 frames while Fig. 5.1(b) only needs 4 active frames per 12 frames, thus the latter improves energy efficiency by 33.3%. In addition, the ratio of sleep frames of M_2 is increased from 75% to 91.7% (for M_1 , it is the same). Obviously, by using one sleep cycle for each MS, it has a great potential for further energy saving of the system and each MS.

This chapter considers the sleep scheduling for multiple MSs under a BS. We are given n MSs M_i , $i = 1..n$ and each M_i has a delay bound for traffic arrivals D_i (frames) and a per frame data arrival rate τ_i (bits/frame). Assuming the available bandwidth per frame of the system to be Ω bits and $\sum_{i=1..n} \tau_i \leq \Omega$, the goal is to assign each M_i a PSC of type II P_i with sleep cycle T_i (frames), listening window T_i^L (frames), and starting frame of the first listening window T_i^S , such that $T_i \leq D_i$ (delay bound), $\tau_i \times T_i \leq T_i^L \times \Omega$ (bandwidth requirement), $i = 1..n$, and the ratio of active frames for the system is minimized.

5.2 Per-MS Sleep Scheduling (PMSS) Method

In an IEEE 802.16e wireless network, the BS is the central of the network, which is responsible to schedule the sleep of the MSs that associate with it. Initially, the MSs

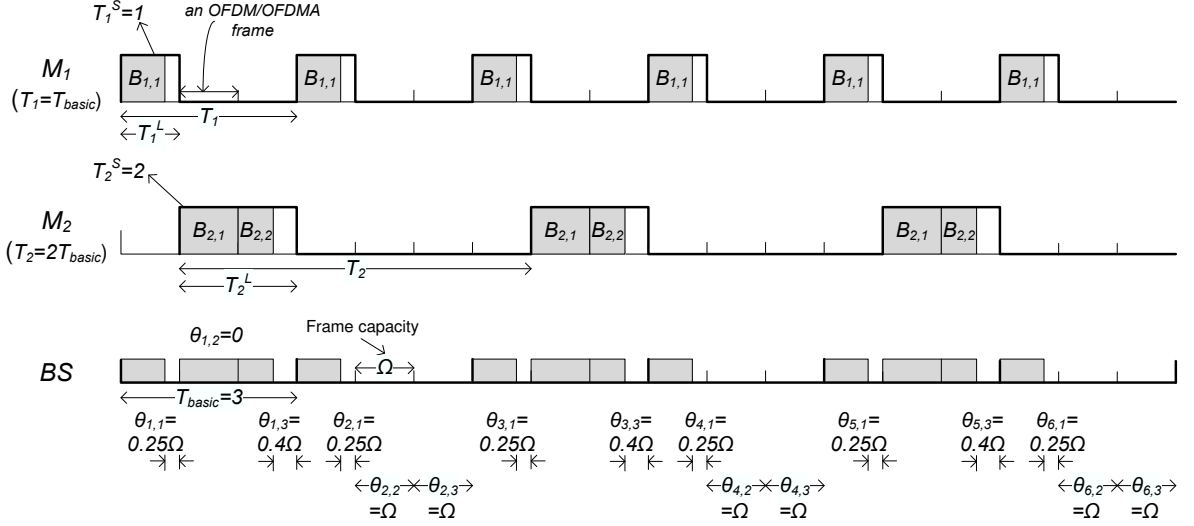


Figure 5.2: Example of the sleep parameters for the MSs and the data structure used in our proposed algorithm.

which intend to enter the sleep mode will send a request to the BS. With all MSs' QoS parameters recorded in the database, the BS will schedule the sleep of the MSs without breaking their QoS requirements. Here we propose a *Per-MS Sleep Scheduling (PMSS)* algorithm to schedule the sleep of MSs, which will determine the following parameters for each M_i , $i = 1..n$: (1) the size of sleep cycle T_i , (2) the size of listening window T_i^L in each sleep cycle, (3) the starting frame T_i^S of the first listening window, and (4) the amount of bandwidth allocation $B_{i,j}$ that is reserved to the j -th active frames of each T_i^L , $j = 1..T_i^L$. After the calculation, the BS will send each MS a response with the four sleep parameters. Upon the receipt of the response, MSs will sleep accordingly.

Basically, our PMSS scheme will maintain a property that the sleep cycle T_i , $i = 2..n$, of each M_i is a positive integer multiple of the previous T_{i-1} . So, we call T_1 as the basic cycle T_{basic} because each T_i , $i = 2..n$ will be the integer multiple of T_1 . We maintain this property is to increase the overlap of the listening windows of MSs such that the the duty cycle of each MS can be reduced and the resource utilization of each frame can be raised. Assuming that T_{basic} is known, our PMSS algorithm involves two steps: (A) determining the sleep cycle T_i of each M_i and (B) scheduling the bandwidth amount $B_{i,j}$, the listening window T_i^L , and the starting frame T_i^S of each M_i . In the end, we will propose a search method to find the best basic cycle T_{basic} so as to derive the most power saving sleep scheduling for the MSs.

PMSS scheme will schedule the sleep of M_i one-by-one and each time the M_i is scheduled to create the least number of active frames for it. To realize this idea, we will maintain a data structure $\theta_{k,\ell}$, $k = 1..T_n/T_{basic}$ and $\ell = 1..T_{basic}$, to record the amount of free resource in the ℓ -th frame of the k -th basic cycle. We use Θ_k to represent the set

of $\{\theta_{k,1}, \theta_{k,2}, \dots, \theta_{k,T_{basic}}\}$. Initially, $\theta_{k,\ell} = \Omega$, In the end of this part, we use Fig. 5.2 to summarize the above sleep parameters T_i, T_i^L, T_i^S , and $B_{i,j}$ for the MSs and data structure $\theta_{k,\ell}$ and π_k maintained by the BS.

5.2.1 Determining T_i of each M_i

In the PMSS scheme, we first sort MSs by their delay bounds such that $D_1 \leq D_2 \leq \dots \leq D_n$. Later on, we will explain the insights why our scheme schedule M_i s in this sequence in the end of the section.

So, in this step, we let $T_1 = T_{basic}$ and set $T_i, i = 2..n$, as follows:

$$T_i = T_{i-1} \times \left\lfloor \frac{D_i}{T_{i-1}} \right\rfloor. \quad (5.1)$$

Actually, the initial T_1 would be set less or equal to D_1 (the assignment of T_1 will be discussed and shown later on). Since $D_{i-1} \leq D_i$ and $T_{i-1} \leq D_{i-1}$, Eq. (5.1) must set T_i as a positive integer multiple of the previous T_{i-1} . Also, Eq. (5.1) implies that $T_i \leq T_{i-1} \times \frac{D_i}{T_{i-1}} = D_i$. So, T_i is guaranteed to meet the delay bound of M_i .

Theorem 5.2.1. *In PMSS, it is guaranteed that each T_i is a positive integer multiple of $T_1 = T_{basic}$, $i = 2..n$, and each M_i 's cycle meets its delay bound, i.e., $T_i \leq D_i, i = 1..n$.*

5.2.2 Scheduling $B_{i,j}, T_i^L$, and T_i^S of each M_i

Recall the data structure $\theta_{k,\ell}, k = 1.. \frac{T_n}{T_{basic}}$ and $\ell = 1..T_{basic}$. We will sequentially schedule $M_i, i = 1..n$, by updating $\theta_{k,\ell}$. Specifically, when M_i is under consideration, we will pick one basic cycle among the first $\frac{T_i}{T_{basic}}$ basic cycles and select one frame in the basic cycle as the starting point of the first listening window for M_i . If M_i cannot get enough resource in the selected basic cycle, we will repeat to reserve resource from the subsequent basic cycles until sufficient resource for M_i is allocated. Among all candidate basic cycles, the one which creates the least increment on the number of active frames is chosen. The detail procedure to place M_i 's demand is illustrated as follows:

a) Calculate the demand \mathfrak{R}_i of M_i per T_i by

$$\mathfrak{R}_i = \tau_i \times T_i. \quad (5.2)$$

b) Since M_i has a period of T_i frames and Θ_k has a period of $\frac{T_{i-1}}{T_{basic}}$ basic cycles after placing $M_1 \sim M_{i-1}$'s demand (to be explained later), we only need to check the first $\frac{T_i}{T_{basic}}$ basic cycles among all $\frac{T_n}{T_{basic}}$ basic cycles to find the best start basic cycle and frame for M_i . Specifically, for $k = 1$ to $\frac{T_i}{T_{basic}}$ with $\theta_{k,T_{basic}} > 0$, we compute a cost function $f(k)$ to represent the active frames incurred to M_i if we place its demand starting from the k -th basic cycle. The demand placement is done as follows:

- i) To make the resource usage in each frame more compact, if $\mathfrak{R}_i - \lfloor \mathfrak{R}_i \rfloor \leq \theta_{k, \tilde{\ell}}$, where $\tilde{\ell}$ is the last frame in the k -th basic cycle with free resource less than Ω , we continuously allocate resource to M_i starting from the $\tilde{\ell}$ -th frame; otherwise, in order to reduce the number of active frames for M_i , we allocate resource to M_i starting from the $(\tilde{\ell} + 1)$ -th frame in π_k which is with free resource Ω .
- ii) If the reserved resource in the k -th frame is not enough for M_i 's requirement \mathfrak{R}_i , we will repeat reserving resource from the subsequent basic cycles. The allocation will examine frames in sequence of the frame index and reserve the free space in each frame until sufficient resource is allocated to M_i .
- c) Let k^* be the index which induces the smallest cost function $f(k)$ in step b. We will place M_i 's demand starting from the k^* -th basic cycle according to above procedure. In case that there is a tie, we will give priority to the one which leaves the least remaining resource in the last frame where M_i 's demand is placed. If there are still more than one basic cycles with priority, choose the one with the least basic cycle index.
- d) Then we set T_i^S to the frame index of the first frame where M_i 's demand is placed and set T_i^L to the number of frames from the first to the last frame where M_i 's demand is placed. Also, $B_{i,j}$ is set accordingly. Finally, we update the remaining free resources in $\theta_{k,\ell}$, $k = 1..n$ and $\ell = 1..T_{basic}$ (note that since the period is T_i , $\theta_{k,\ell} = \theta_{k+m\frac{T_i}{T_{basic}},\ell}$, $m = 1..(\frac{T_n}{T_i} - 1)$).

In our scheme, we guarantee that Θ_k has a period of $\frac{T_i}{T_{basic}}$ basic cycles after placing $M_1 \sim M_i$'s demands by forcing T_i as a positive integer multiple of the previous T_{i-1} . This can easily be proved by induction. It is true when $i = 1$. Assuming it is also true for $i = j - 1$, we're now going to prove it is also true for $i = j$. Before placing M_j 's demand, we know Θ_k has a period of $\frac{T_{j-1}}{T_{basic}}$ basic cycles. This is equivalent to that Θ_k has a period of $\frac{T_j}{T_{basic}}$ basic cycles. According to step b of the above procedure, we will place M_j 's demand starting from the k^* -th basic cycle and allocate resources to C_j from the subsequent basic cycles until sufficient resource is allocated. Since the period of M_j is T_j , $\theta_{k,\ell} = \theta_{k+m\frac{T_j}{T_{basic}},\ell}$, $m = 1..(\frac{T_n}{T_j} - 1)$. So, Θ_k has a period of $\frac{T_i}{T_{basic}}$ basic cycles after placing $M_1 \sim M_i$'s demands.

Example 5.2.1. Fig. 5.3 shows an example of step B. There are 5 MSs M_1, M_2, M_3, M_4 , and M_5 with sleeping cycles of $T_1 = T_{basic}$, $T_2 = 2T_{basic}$, $T_3 = 2T_{basic}$, $T_4 = 2T_{basic}$, and $T_5 = 4T_{basic}$ and required resources per cycle of $\mathfrak{R}_1 = 0.5\Omega$, $\mathfrak{R}_2 = 1.25\Omega$, $\mathfrak{R}_3 = 0.4\Omega$, $\mathfrak{R}_4 = 0.4\Omega$, and $\mathfrak{R}_5 = 2.5\Omega$, respectively, where $T_{basic} = 3$ frames. Initially, $\theta_{k,\ell} = \Omega$ for $k = 1..4$ and $\ell = 1..3$. Then, each M_i is scheduled as follows. For M_1 , we can only set

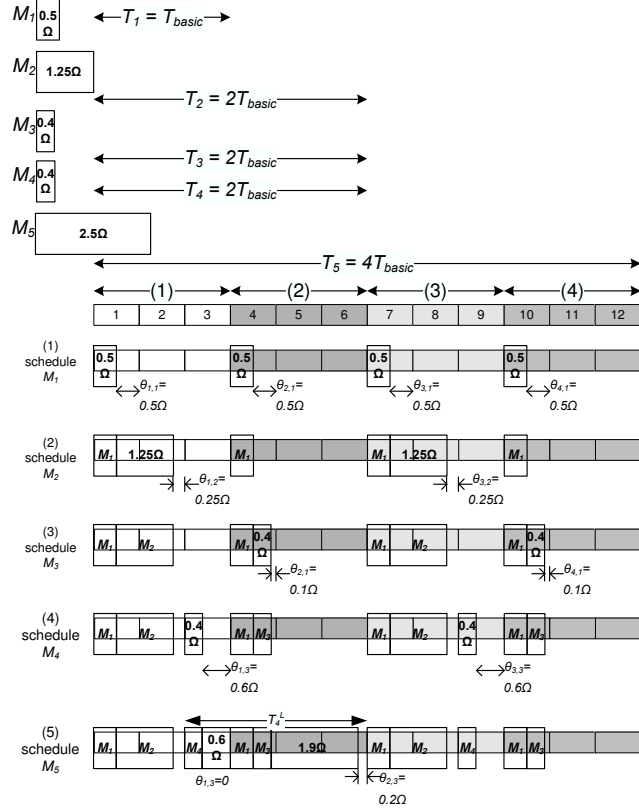


Figure 5.3: Example of scheduling $B_{i,j}$, T_i^L , and T_i^S of four MSs.

$k^* = 1$. Then, the BS reserves $\mathfrak{R}_1 = 0.5\Omega$ resource for M_1 in every basic cycle as shown in Fig. 5.3(1) and set $B_{1,1} = 0.5\Omega$, $T_1^S = 1$, and $T_1^L = 1$; so $\theta_{1,1} = \theta_{2,1} = \theta_{3,1} = \theta_{4,1} = 0.5\Omega$ and $\theta_{k,\ell} = \Omega$ for $k = 1..4$ and $\ell = 2, 3$. For M_2 , its k^* can be 1 or 2 and allocating \mathfrak{R}_2 by starting from any of the two basic cycles are the same. So we select $k^* = 1$. Since $\mathfrak{R}_2 - \lfloor \mathfrak{R}_2 \rfloor = 0.25\Omega \leq \theta_{1,1} = 0.5\Omega$, M_2 is reserved resource from the first frame of the first basic cycle. After the allocation, shown as Fig. 5.3(2), we set $B_{2,1} = 0.5\Omega$, $B_{2,2} = 0.75\Omega$, $T_2^S = 1$, and $T_2^L = 2$ and update $\theta_{1,1} = \theta_{3,1} = 0$ and $\theta_{1,2} = \theta_{3,2} = 0.25\Omega$. For M_3 , choosing $k^* = 1$ or 2 both would create the same number of active frames (i.e., $f(1) = f(2) = 1$), but setting $k^* = 2$ would leave less remaining resource in the last allocated frame (i.e., 0.1Ω). So we set $k^* = 2$ and update $B_{3,1} = 0.4\Omega$, $T_3^S = 4$, and $T_3^L = 1$, as shown in Fig. 5.3(3). Then update $\theta_{2,1} = \theta_{4,1} = 0.1\Omega$. For M_4 , setting $k^* = 1$ or 2 both would create the same number of active frames (i.e., $f(1) = f(2) = 1$) and leave the same remaining resource in the last frame. So we choose $k^* = 1$ and set $B_{4,1} = 0.4\Omega$, $T_4^S = 3$, and $T_4^L = 1$, as shown in Fig. 5.3(4). Then we update $\theta_{1,3} = \theta_{3,3} = 0.6\Omega$. For M_5 , choosing $k^* = 1$ and 3 would add the least number of active frames (i.e., $f(1) = f(3) = 4 < f(2) = f(4) = 5$). Since $k^* = 1$ and 3 will both leave the same remaining resource in the last frame, we pick $k^* = 1$. So we set $B_{5,1} = 0.6\Omega$, $B_{5,2} = 0.1\Omega$, $B_{5,3} = \Omega$, $B_{5,4} = 0.8\Omega$, $T_5^S = 3$, and $T_5^L = 4$, as shown in Fig. 5.3(5). Then update $\theta_{1,3} = \theta_{2,1} = \theta_{2,2} = 0$ and $\theta_{2,3} = 0.2\Omega$.

5.2.3 Determining T_{basic}

In the part, we present how to determine the basic cycle T_{basic} . Since $T_{basic} = T_1$ and T_1 must be guaranteed $T_1 \leq D_1$, $1 \leq T_{basic} \leq D_1$. This guarantees Theorem 5.2.1 to be true. To get the most power conserving T_{basic} , for $T_{basic} = 1$ to D_1 , we compute an energy function $e(T_{basic})$ to represent the ratio of active frames of the system if T_{basic} is used. Then, the basic cycle T_{basic}^* which induces the least energy function $e(T_{basic})$ will be chosen as the final basic cycle. For each sleep scheduling result by a given T_{basic} , $e(T_{basic})$ can be derived by

$$e(T_{basic}) = \frac{\sum_{i=1}^n \left(\frac{T_n}{T_i} \times T_i^L \right)}{T_n} \quad (5.3)$$

5.3 Experimental Results

We have developed a simulator by C++ to verify the effectiveness of our PMSS scheme. Unless otherwise stated, the following assumptions are made in our simulation. The number of MSs is ranged from 5 to 45. Each MS M_i has a data rate τ_i of 1000 ~ 3000 bits/frame and delay bound D_i 10 ~ 200 frames, where 1000 is the minimum data rate, 3000 is the maximum data rate, 10 is the minimum delay bound, and 200 is the maximum delay bound of the MS. The available bandwidth per frame of the system is $\Omega = 80000$ bits (16Mbps) and the length of an OFDM/OFDMA frame is set to 5 ms[30]. We consider two performance metrics: (i) *active ratio*: the ratio of active frames for the system and (ii) *fail-to-sleep probability*: the ratio of failure to schedule MSs' sleep. We will compare our PMSS against the MMPS-FC (Multiple MSs Power-saving Scheduler with Fragment Collection) and MMPS-BF (Multiple MSs Power-saving Scheduler with Boundary Free) schemes in[24].

Effects of n

In this experiment, we study the effect of n on the active ratio and fail-to-sleep probability. Fig. 5.4(a) shows the active ratio decreases when n increases. Our PMSS almost always performs the best in all three schemes, except at $n = 40$, MMPS-FC performs better than our PMSS. However, when $n = 40$, the fail-to-sleep probability of MMPS-FC is almost 100% (Fig. 5.4(b)). Fig. 5.4(b) shows the fail-to-sleep probability increases when n increases. MMPS-BF and our PMSS schemes perform the same and the best in the fail-to-sleep probability. The fail-to-sleep probabilities of the two schemes is zero when $n < 40$. For MMPS-FC, it can 100% successfully schedule MSs into sleep when $n < 25$.

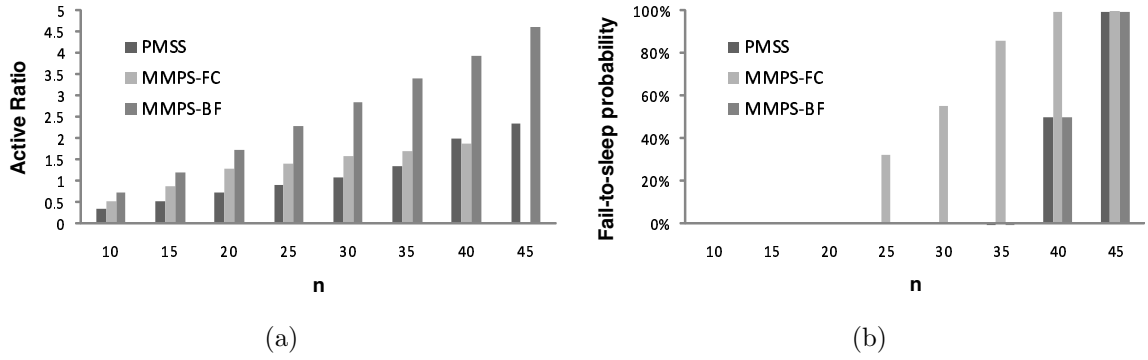


Figure 5.4: Effects of number of MSs on (a) active ratio and (b) fail-to-sleep probability.

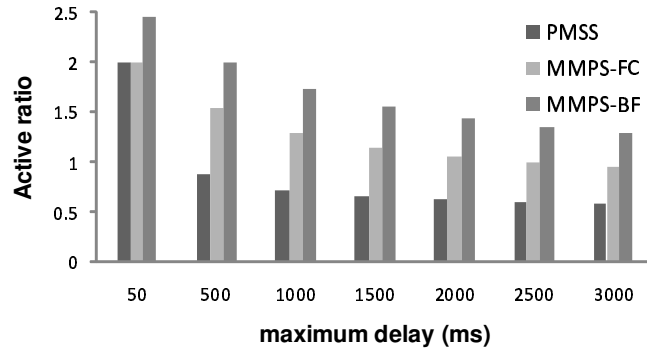


Figure 5.5: Effects of maximum delay on active ratio.

Effects of Maximum Delay Bound

Then, we investigate the effect of maximum delay bound on the active ratio by fixing $n = 20$. Fig. 5.5 shows the active ratio decreases when the maximum delay bound increases. Our PMSS performs the best in all three schemes. For the three schemes, our PMSS benefits the most when the maximum delay bound is increased from 50 ms to 3000 ms (70%); for MMPS-FC and MMPS-BF, the improvement is 52% and 47%, respectively. This is because our scheme can more accurately capture the traffic characteristics of MSs.

Effects of the Difference between Minimum and Maximum Data Rate

We study the effect of the difference between minimum and maximum data rate on the active ratio by fixing $n = 20$. Basically, active ratio increases when the difference between minimum and maximum data rate increases (as shown in Fig. 5.6). In all three schemes, the difference between minimum and maximum data rate has the least effect on our scheme (2.8%) and has the most effect on MMPS-BF (6.3%).

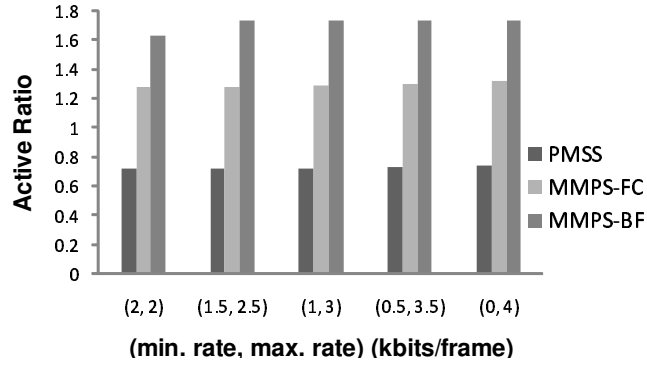


Figure 5.6: Effects of the difference between minimum and maximum data rate on active ratio.

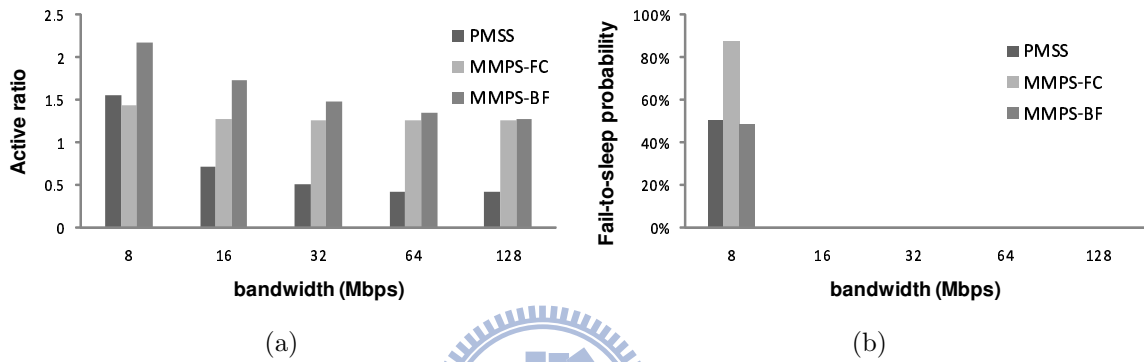


Figure 5.7: Effects of system bandwidth on (a) active ratio and (b) fail-to-sleep probability.

Effects of System Bandwidth

In this experiment, we investigate the effect of system bandwidth on the active ratio and fail-to-sleep probability by fixing $n = 20$. Fig. 5.7(a) shows the active ratio decreases when system bandwidth increases. Our PMSS outperforms other two schemes except when the system bandwidth is 8Mbps. When the system bandwidth is 8Mbps, MMPS-FC performs the best but its fail-to-sleep probability is much higher (88%) than other two schemes (about 50%). For the three schemes, our PMSS benefits the most when system bandwidth is increased from 8Mbps to 128Mbps (73%); for MMPS-FC and MMPS-BF, the improvement is 13% and 42%, respectively.

Chapter 6

Conclusions and Future Directions

This dissertation contains three major works. In the first work, we propose a mobile broadband network architecture. In the second work, we discuss the resource management issue for the mobile broadband networks. In the last work, the power management issue for the mobile broadband networks is studied. In the following, we summarize this dissertation.

Chapter 3 includes our first two works. First, we have proposed a SIP-based mobile broadband network architecture which can support broadband networking services for a group of users who roam together. With multiple wireless interfaces, a SIP-MNG can provide dynamic bandwidth to internal users based on their bandwidth requirements. Also, by allowing multiple sessions to share one interface, our system can help users or public transportation operators to save Internet access fees. To allow the SIP-MNG to stay off-line when there is no calling activity, we have proposed a push mechanism to “wake up” the SIP-MNG when necessary. The push approach can help to save call charges, network resource, and energy consumption, while maintaining global reachability of users. By interpreting SIP signaling, our RM and CAC mechanisms inside the SIP-MNG can guarantee QoS for users. We develop a prototype and some experimental results are presented. For IEEE 802.11, WCDMA, and PHS networks, we demonstrated that it is feasible to allow multiple stations to share one interface. It is also shown that, by cellular interfaces, the call setup time and handoff delay are longer than that by 802.11 interfaces, because connecting to the Internet via a cellular interface has to go through more networks. Then, to reduce the effects of handoff and physical rate adaptation on QoS of users and mobile broadband networks, we have further proposed a CAC and a RA mechanisms over IEEE 802.11e multi-rate wireless networks. By upgrading/degrading resources allocated to calls, we can make better use of the network resource. We have also derived an analytical model to evaluate our system with multi-level QoS support. Three performance

metrics, blocking probability, dropping probability, channel utilization, have been derived. Our numerical analysis shows the importance of CAC and RA mechanisms, especially when user mobility is high and fairness is important. Our simulation results also support our conclusions. Observing that a handoff in a wireless network with complete security support consume a long time, we propose a Dynamic Tunnel Establishing procedure and a seamless post-handoff method to provide mobile users the seamless handoff and continuous network connectivity over DHCP-based IEEE 802.11 wireless networks which support IEEE 802.11i. During a handoff, by using our proposed approach, the Probe-and-Decision phase can be reduced to less than 20–30 ms by adopting one of existing schemes [57, 23, 46, 63], and the three remaining phases, Execution, DHCP, and Upper Layer Adjustment phases, can be hidden by the tunneling services. Therefore, for a roaming MS, the continuous disconnected period with the Internet can be guaranteed to be less than 50 ms. So, a seamless handoff is concluded. In addition, the proposed method provides the same security level as the original IEEE 802.11i standard because, during a handoff, the moving MS is allowed to access the Internet only when it is permitted by the old AP and the old AP is trusted. Moreover, if there's any improvement or change for the authentication and encryption methods, our seamless handoff mechanism still can work correctly because it doesn't involve any modification to the authentication and encryption methods.

In Chapter 4, we have proposed three power saving class management schemes for a BS-MS pair in IEEE 802.16e wireless networks. They are FD, PSS-DB, and PSS-PI schemes. All these sleep scheduling methods conform to the sleep mechanism defined in IEEE 802.16e and easy to implement. The three schemes consider the maximum delay constraint, packet inter-arrival time, and data rate of connections to determine the parameters of PSCs. Multiple PSCs of type II are used to capture the sleep-active behavior contributed by real-time flows. One PSC of type I is used to handle non-real-time flows. For FD scheme, we have also proposed an earliest-next-bandwidth-first scheduler, which can guarantee the real-time flows' delay constraints. Different from FD scheme, PSS-DB and PSS-PI assign each flow a PSC, so we can schedule the packet delivery by just following the active time of PSCs. In addition, for each real-time connection, PSS-DB considers the delay bound to assign the sleeping cycle while PSS-PI uses the packet inter-arrival time to assign the sleeping cycle. Furthermore, we suggest to use the PSS-PI when the packet size is small and the total bandwidth requirement of the MS is limited compared to B . Otherwise, we can adopt PSS-DB. We also prove that deciding whether a given scheduling problem is solvable can be reduced to a maximum matching problem, which can always be solved in polynomial time. Simulation results show that, compared to the single PSC solution, our schemes can save the MS's energy even when there are

many real-time flows coexist while keep bandwidth utilization high under real-time flows' delay constraints.

In Chapter 5, we propose a per-MS sleep scheduling scheme for multiple MSs under a BS in IEEE 802.16e wireless networks. PMSS scheme minimizes the overall power consumption of a MAN (metropolitan area network) while the QoS of each MS can be guaranteed. Besides, it conforms to the sleep mechanism defined in IEEE 802.16e and easy to implement. Compared to the previous work, our approach assigns and schedules type II PSCs for each MS by considering each of their QoS characteristics such that the sleep scheduling can more accurately capture each MS's QoS requirement. This leads to each MS can sleep more and the overall active frames of multiple MSs under a BS is significantly reduced.

Based on the results presented above, several issues worth further investigation are summarized as follows.

- It deserves to further test our designed SIP-based mobile broadband network on a moving cars and public transportation to evaluate the performance and then realize the mobile broadband network in the real world.
- In our current push mechanism, the wireless interface reconnection time takes the largest part of the time. Actually, there have be some research works focusing on similar problem [67, 41]. We believe that, with further study and improvement, it is able to shorten the reconnection time and make the push mechanism more practical.
- Our current cross-layer resource management mechanism focus on VoIP and IEEE 802.11e. In the future, we can further discuss how to apply the idea on more general kinds of real-time applications and more kinds of wireless network technologies, such as IEEE 802.16.
- It deserves to further discuss how to extend the post-handoff concept in more complicated wireless networks such as heterogeneous networks.
- Our current PSC management schemes assume fixed bit rate real-time flows. Although they are also applicable to variable bit rate real-time flows, the performance of power consumption and resource utilization must degrade. In this case, a further design for PSC management is needed. This deserves further study. Furthermore, considering that a radio has to consume additional power when switching from sleep to active or active to sleep, it is worth to further improve the PSC management schemes by taking this factor into consideration.

Bibliography

- [1] IEEE Std 802.11-1997 Information Technology- telecommunications And Information exchange Between Systems-Local And Metropolitan Area Networks-specific Requirements-part 11: Wireless Lan Medium Access Control (MAC) And Physical Layer (PHY) Specifications. Nov. 1997.
- [2] IEEE Std 802.11f, IEEE Trial-Use Recommended Practice for Multi-Vendor Access Point Interoperability via an Inter-Access Point Protocol Across Distribution Systems Supporting IEEE 802.11 Operation. July 2003.
- [3] Recommended Practice for Multi-Vendor Access Point Interoperability via an Inter-Access Point Protocol Across Distribution Systems Supporting IEEE 802.11 Operation. *IEEE Draft 802.1f/Final Version*, Jan 2003.
- [4] IEEE Std 802.11e-2005, Part 11: Wireless LAN Medium Access Control(MAC) and Physical Layer(PHY) specifications. Amendment 8: Medium Access Control(MAC) Quality of Service(QoS) Enhancements. Nov. 2005.
- [5] IEEE Std. 802.11i. IEEE Standard for Information technology- Telecommunications and information exchange between systems- Local and metropolitan area networks- Specific requirements Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications Amendment 6: Medium Access Control (MAC) Security Enhancements. Nov. 2004.
- [6] B. Aboba. Fast Handoff Issues. IEEE-03-155r0-I. *IEEE 802.11 Working Goup*, March 2003.
- [7] H. Balakrishnan, V. N. Padmanabhan, S. Seshan, and R. H. Katz. A Comparison of Mechanisms for Improving TCP Performance over Wireless Links. *IEEE/ACM Transactions on Networking*, 5(6):756–769, Dec. 1997.
- [8] D. Chen, A. K. Elhakeem, and X. Wang. A Novel Call Admission Control in Multi-Service Wireless LANs. In *In Proc. of the 3rd Int'l Symposium on Modeling and*

- Optimization in Mobile, Ad Hoc, and Wireless Networks (WIOPT 2005)*, pages 119–128, Apr. 2005.
- [9] T.-C. Chen, J.-C. Chen, and Y.-Y. Chen. Maximizing Unavailability Interval for Energy Saving in IEEE 802.16e Wireless MANs. *IEEE Transactions on Mobile Computing*, 8(4):475–487, Apr. 2009.
- [10] C.-C. Chiang, H.-K. Wu, W. Liu, and M. Gerla. Routing in Clustered Multihop, Mobile Wireless Networks with Fading Channel. *Proc. IEEE Singapore Int. Conf. on Networks*, pages 197–211, Apr. 1997.
- [11] C.-T. Chou and K.-G. Shin. Analysis of Adaptive Bandwidth Allocation in Wireless Networks with Multilevel Degradable Quality of Service. *IEEE Trans. on Mobile Computing*, 3(1):5–17, Jan./Feb. 2004.
- [12] T. H. Cormen, C. E. Leiserson, and R. L. Rivest. *Introduction to Algorithms*. MIT Press, 2001.
- [13] V. Devarapalli, R. Wakikawa, A. Petrescu, and P. Thubert. Network Mobility Basic Support Protocol. *IETF RFC 3963*, Jan. 2005.
- [14] T. Ernst. Network Mobility Support Goals and Requirements. *IETF DRAFT: draft-ietf-nemo-requirements-06*, Nov. 2006.
- [15] T. Ernst and H.-Y. Lach. Network Mobility Support Terminology. *IETF DRAFT: draft-ietf-nemo-terminology-06*, Nov. 2006.
- [16] T. Ernst, K. Uehara, and K. Mitsuya. Network Mobility from the InternetCAR Perspective. In *Proc. of the 17th IEEE Int'l Conf. on Advanced Inf. Networking and Appl.*, pages 19–25, Mar. 2003.
- [17] P. Ferguson and D. Senie. Network Ingress Filtering: Defeating Denial of Service Attacks which Employ IP Source Address Spoofing. *IETF RFC 2267*, Jan. 1998.
- [18] A. Floris, L. Tosetti, and L. Veltri. Solutions for Mobility Support in DHCP-based Environments. In *Proceedings of IEEE ICC'03*, pages 1043–1047, 2003.
- [19] C.-H. Gan and Y.-B. Lin. An Effective Power Conservation Scheme for IEEE 802.11 Wireless Networks. *IEEE Transactions on Vehicular Technology*, 58(4):1920–1929, May 2009.
- [20] M. Handley, V. Jacobson, and C. Perkins. SDP: Session Description Protocol. *IETF RFC 4566*, July 2006.

- [21] U. Herzog, L. Woo, and K. Chandy. Solution of queueing problems by a recursive technique. *IBM J. Res. Dev.*, 19:295–300, May 1975.
- [22] C.-M. Huang, C.-H. Lee, and J.-R. Zheng. A Novel SIP-Based Route Optimization for Network Mobility. *IEEE Journal on Selected Areas in Communications*, 24(9):1682–1691, Sep. 2006.
- [23] P.-J. Huang, Y.-C. Tseng, and K.-C. Tsai. A Fast Handoff Mechanism for IEEE 802.11 and IAPP Networks. In *VTC2006-Spring*, volume 2, pages 966–970, 2006.
- [24] S.-C. Huang, R.-H. Jan, and C. Chen. Energy Efficient Scheduling with QoS Guarantee for IEEE 802.16e Broadband Wireless Access Networks. In *Proc. of the 2007 International conference on Wireless Communications and Mobile Computing (IWCMC 2007)*, pages 547–552, Aug. 2007.
- [25] IEEE Std. 802.16-2004. IEEE Standard for Local and Metropolitan Area Networks Part 16: Air Interface for Fixed Broadband Wireless Access Systems, Oct. 2004.
- [26] IEEE Std. 802.16e-2005. Part 16: Air Interface for Fixed and Mobile Broadband Wireless Access Systems - Amendment 2: Physical and Medium Access Control Layers for Combined Fixed and Mobile Operation in Licensed Bands, Feb. 2006.
- [27] D. Johnson, C. Perkins, and J. Arkko. Mobility Support in IPv6. *IETF RFC 3775*, Jun. 2004.
- [28] G. Karlsson. Asynchronous transfer of video. *IEEE Communication Magazine*, 11:118–126, Aug 1996.
- [29] H.-S. Kim, S.-H. Park, C.-S. Park, J.-W. Kim, and S.-J. Ko. Selective Channel Scanning for Fast Handoff in Wireless LAN using Neighbor Graph. In *the Int'l Technical Conference on Circuits/Systems, Computers and Communications (ITC-CSCC2004)*, 2004.
- [30] H.-S. Kim and S. Yang. Tiny MAP: An efficient MAP in IEEE 802.16/WiMAX broadband wireless access systems. *Computer Communications*, 30(9):2122–2128, June 2007.
- [31] Y.-C. Kim, D.-E. Lee, B.-J. Lee, Y.-S. Kim, and B. Mukherjee. Dynamic Channel Reservation based on Mobility in Wireless ATM. In *Proc. of IEEE wmATM'99*, pages 100–106, 1999.
- [32] R. Krashinsky and H. Balakrishnan. Minimizing Energy for Wireless Web Access with Bounded Slowdown. In *Proc. of ACM Mobicom2002*, pages 119–130, Sep. 2002.

- [33] H.-Y. Lach, C. Janneteau, and A. Petrescu. Network Mobility in Beyond-3G Systems. *IEEE Communications Magazine*, 41(7):52–57, Jul. 2003.
- [34] H.-P. Lin, S.-C. Huang, and R.-H. Jan. A power-saving scheduling for infrastructure-mode 802.11 wireless LANs. *Computer Communications*, 29(17):3483–3492, Nov. 2006.
- [35] Y.-B. Lin, W.-E. Chen, and C.-H. Gan. Effective VoIP Call Routing in WLAN and Cellular Integration. *IEEE Communications Letters*, 9(10):874–876, Oct. 2005.
- [36] Y.-B. Lin, Y.-C. Lo, and C.-H. Rao. A Push Mechanism for GPRS Supporting Private IP Addresses. *IEEE Communications Letters*, 7(1):24–26, Jan. 2003.
- [37] A. McAuley and K. Manousakis. Self-Configuring Networks. In *1st Century Military Communications Conference, MILCOM 2000, Proceedings*, volume 1, pages 315–319, 2000.
- [38] A. Mishra, M. Shin, and W. Arbaugh. An Empirical Analysis of the IEEE 802.11 MAC Layer Handoff Process. *ACM SIGCOMM Computer Comm. Rev.*, 33(2):93–102, April 2003.
- [39] N. Nasser. Adaptability Enhanced Framework for Provisioning Connection-level QoS in Multimedia Wireless Networks. *IEEE and IFIP International Conference on Wireless and Optical Communications Networks (WOCN)*, pages 275–279, Mar 2005.
- [40] netfilter. <http://www.netfilter.org/>.
- [41] S. Park, P. Kim, M. Lee, and Y. Kim. *Parallel and Distributed Computing: Applications and Technologies*, chapter Fast Address Configuration for WLAN, pages 396–400. Springer Berlin/Heidelberg, 2004.
- [42] E. Perera, V. Sivaraman, and A. Seneviratne. Survey on Network Mobility Support. *ACM SIGMOBILE Mobile Computing and Communications Review*, 8(2):7–19, Apr. 2004.
- [43] C. Perkins. IP Mobility Support for IPv4. *IETF RFC 3344*, Aug. 2002.
- [44] C. E. Perkins and P. Bhagwat. Highly Dynamic Destination-Sequenced Distance-Vector Routing (DSDV) for Mobile Computers. *Proc. of the ACM SIGCOMM*, 234–244, 1994.
- [45] D. Pong and T. Moors. Call admission control for IEEE 802.11 contention access mechanism. In *Proc. of IEEE GLOMECOM*, 1:174–178, 2003.

- [46] I. Ramani and S. Savage. SyncScan: Practical Fast Handoff for 802.11 Infrastructure Networks. In *INFOCOM2005*, volume 1, pages 675–684, March 2005.
- [47] C.-H. Rao, D.-F. Chang, and Y.-B. Lin. iSMS: An Integration Platform for Short Message Service and IP Networks. *IEEE Networks*, 15:48–55, Mar./Apr. 2001.
- [48] C. Rigney, S. Willens, A. Rubens, and W. Simpson. Remote Authentication Dial In User Service (RADIUS). *RFC 2865*, June 2000.
- [49] A. B. Roach. Session Initiation Protocol (SIP)-Specific Event Notification. *IETF RFC 3265*, June 2002.
- [50] J. Rosenberg. Interactive Connectivity Establishment (ICE): A Methodology for Network Address Translator (NAT) Traversal for Offer/Answer Protocols. *IETF DRAFT: draft-ietf-mmusic-ice-15*, Mar. 2007.
- [51] J. Rosenberg, D. Drew, and H. Schulzrinne. Getting SIP through Firewalls and NATs. *IETF DRAFT: draft-rosenberg-sip-firewalls-00*, Feb. 2000.
- [52] J. Rosenberg, H. Schulzrinne, G. Camarillo, A. Johnston, J. Peterson, R. Sparks, M. Handley, and E. Schooler. SIP: Session Initiation Protocol. *IETF RFC 3261*, June 2002.
- [53] J. Rosenberg, J. Weinberger, C. Huitema, and R. Mahy. STUN - Simple Traversal of User Datagram Protocol (UDP) Through Network Address Translators (NATs). *IETF RFC 3489*, Mar. 2003.
- [54] J. Rosengerg, J. Peterson, H. Schulzrinne, and G. Camarillo. Best Current Practices for Third Party Call Control (3pcc) in the Session Initiation Protocol (SIP). *IETF RFC 3725*, Apr. 2004.
- [55] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson. RTP: A Transport Protocol for Real-Time Applications. *IETF RFC 3550*, July 2003.
- [56] SIP Express Router (ser). <http://www.iptel.org/ser/>.
- [57] M. Shin, A. Mishra, and W. A. Arbaugh. Improving the Latency of 802.11 Handoffs using Neighbor Graphs. In *Proc. of the 2nd int'l conference on Mobile Systems, Applications, and Services (MobiSys2004)*, pages 70–83, June 2004.
- [58] R. Sparks. The Session Initiation Protocol (SIP) Refer Method. *IETF RFC 3515*, Apr. 2003.

- [59] P. Srisuresh and K. Egevang. Traditional IP Network Address Translator (Traditional NAT). *IETF RFC 3022*, Jan 2001.
- [60] J. A. Stine and G. De Ceciana. Improving energy efficiency of centrally controlled wireless data networks. *ACM/Baltzer Wireless Networks*, 8:681–700, 2002.
- [61] S.-L. Tsao and Y.-L. Chen. Energy-efficient packet scheduling algorithms for real-time communications in a mobile WiMAX system. *Computer Communications*, 31(10):2350–2359, June 2008.
- [62] S.-L. Tsao and E.-C. Cheng. PIANO: A power saving strategy for cellular/VoWLAN dual-mode mobiles. *Wireless Networks*, 14(5):683–698, Oct. 2008.
- [63] C.-C. Tseng, K.-H. Chi, M.-D. Hsieh, and H.-H. Chang. Location-based Fast Handoff for 802.11 Networks. *IEEE Comm. Letters*, 9(4):304–306, Apr. 2005.
- [64] H.-L. Tseng, Y.-P. Hsu, C.-H. Hsu, P.-H. Tseng, and K.-T. Feng. A Maximal Power-Conserving Scheduling Algorithm for Broadband Wireless Networks. In *Proc. of IEEE WCNC 2008*, pages 1877–1882, Mar. 2008.
- [65] International Telecommunication Union. General Characteristics of International Telephone Connections and International Telephone Circuits. ITU-TG.114. 1988.
- [66] F. Vakil, A. Dutta, and J. C. Chen. Supporting Mobility for Multimedia with SIP. *draft-itsumo-sipping-mobility-multimedia-01.txt*, July 2001.
- [67] H. Velayos and G. Karlsson. Techniques to Reduce the IEEE 802.11b Handoff Time. In *IEEE ICC*, volume 7, pages 3844–3848, June 2004.
- [68] P. Wang, H. Jiang, and W. Zhuang. Capacity Improvement and Analysis for Voice/Data Traffic over WLANs. *IEEE Transactions on Wireless Communications*, 6(4):1530–1541, Apr. 2007.
- [69] W. Wang, S.-C. Liew, and V. O. K. Li. Solutions to Performance Problems in VoIP Over a 802.11 Wireless LAN. *IEEE Transactions on Vehicular Technology*, 54(1):366–384, Jan. 2005.
- [70] W. Ye, J. Heidemann, and D. Estrin. Medium access control with coordinated adaptive sleeping for wireless sensor networks. *IEEE/ACM Transactions on Networking*, 12(3):493–506, June 2004.
- [71] L. Zan, J. Wang, and L. Bao. Personal AP Protocol for Mobility Management in IEEE 802.11 Systems. In *The 2nd Annual International Conference on Mobile and*

Ubiquitous Systems: Networking and Services (MOBIQUITOUS), San Diego, CA, July 17-21, 2005.

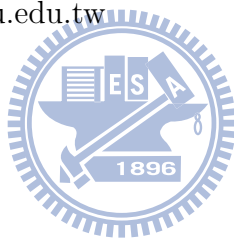
- [72] H. Zhai, X. Chen, and Y. Fang. A call admission and rate control scheme for multimedia support over IEEE 802.11 wireless LANs. *Wireless Networks*, 12(4):451–463, July 2006.



Curriculum Vitae

Jen-Jee Chen was born in Kaohsiung, Taiwan, in 1978, and grew up in Hualien, Taiwan. He received his B.S. and M.S. degrees in Computer Science and Information Engineering from the National Chiao-Tung University, Taiwan, in 2001 and 2003, respectively. He was a Visiting Scholar at the University of Illinois at Urbana-Champaign during the 2007-2008 academic year. He is currently a Ph.D. candidate in the Department of Computer Science, National Chiao-Tung University, Taiwan. His research interests include wireless communications and networks, personal communication service, cross-layer design, network performance modeling and analysis, and voice over IP.

Email address: chencz@cs.nctu.edu.tw



Publication List

Journal Papers

1. Y.-B. Lin, W.-R. Lai, and J.-J. Chen, “Effects of Cache Mechanism on Wireless Data Access,” *IEEE Trans. on Wireless Communications*, 2(6): 1247-1258, Nov. 2003. (SCI, EI)
2. J.-J. Chen, Y.-C. Tseng, and H.-W. Lee, “A Seamless Handoff Mechanism for DHCP-Based IEEE 802.11 WLANs,” *IEEE Communications Letters*, 11(6): 665-667, Aug. 2007. (SCI, EI)
3. Y.-C. Tseng, J.-J. Chen, and Y.-L. Cheng, “Design and Implementation of A SIP-Based Mobile and Vehicular Wireless Network with Push Mechanism,” *IEEE Trans. On Vehicular Technology*, 56(6): 3408-3420, Nov. 2007. (SCI, EI)
4. P.-Y. Wu, J.-J. Chen, Y.-C. Tseng, and H.-W. Lee, “Design of QoS and Admission Control for VoIP Services over IEEE 802.11e WLANs,” *Journal of Information Science and Engineering*, 24(4): 1003-1022, Jul. 2008. (SCIE, EI)

Conference Papers

1. J.-J. Chen, Y.-L. Cheng, Y.-C. Tseng, and Q. Wu, “A Push Mechanism for an Ad-Hoc Network-Based Mobile VoIP Service Platform,” *Mobile Computing Workshop 2006*, Taichung Taiwan, March 31, 2006.
2. J.-J. Chen, Y.-L. Cheng, Y.-C. Tseng, and Q. Wu, “A Push-Based VoIP Service for an Internet-Enabled Mobile Ad Hoc Network,” *IEEE Asia Pacific Wireless Communications Symp. (APWCS'06)*, 2006.
3. J.-J. Chen, L. Li, and Y.-C. Tseng, “Integrating SIP and IEEE 802.11e to Support Handoff and Multi-grade QoS for VoIP Applications,” *ACM International Workshop on QoS and Security for Wireless and Mobile Networks (Q2SWinet'06)*, 2006.

4. J.-J. Chen, Y.-C. Tseng, and H.-W. Lee, “A Seamless Handoff Mechanism for IEEE 802.11 WLANs Supporting IEEE 802.11i Security Enhancements,” *IEEE Asia Pacific Wireless Communications Symp. (APWCS’07)*, 2007.
5. J.-J. Chen, S.-L. Wu, and S.-W. Wang, “Power Saving Class Management for Energy Saving in IEEE 802.16e Wireless Networks,” *The 10th Int’l Conference on Mobile Data Management: Systems, Services and Middleware (MDM’09)*, May 2009, pages 490 – 495.
6. J.-M. Liang, J.-J. Chen, Y.-C. Wang, and Y.-C. Tseng, “Priority-Based Scheduling Algorithm for Downlink Traffics in IEEE 802.16 Networks,” *IEEE Asia Pacific Wireless Communications Symp. (APWCS’09)*, Aug. 2009.
7. S.-L. Wu, J.-J. Chen, and J.-T. Chang, “Cell-centric Area-related Location Area Planning for PCS Networks,” *The 5th Workshop on Wireless Ad Hoc and Sensor Networks (WASN’09)*, Sep. 2009.

Book Chapters

1. J.-J. Chen and Y.-C. Tseng, “Fast Handoff Mechanisms for IEEE 802.11 networks,” *Medium Access Control in Wireless Networks*, Edited by Hongyi Wu (Univ. of Louisiana) and Yi Pan (Georgia State Univ.), NOVA Publications, 2008 (ISBN: 1-60021-944-6).

Patents

1. S.-L. Wu, J.-J. Chen, S.-W. Wang, and Y.-C. Tseng, “An Energy Efficient Multiple Power Saving Class-based Sleep Scheduling Protocol for IEEE 802.16e Wireless Networks,” Taiwan (pending), owned by CGU.
2. J.-J. Chen, Y.-C. Tseng, and H.-W. Lee, “Handoff Method of Mobile Device Utilizing Dynamic Tunnel,” USA (pending), Taiwan (pending), China (pending), owned by Zyxel Communications.

Submitted Journal Papers

1. Y.-C. Tseng, J.-J. Chen, and Y.-C. Yang, “Managing Power Saving Classes in IEEE 802.16e Mobile WiMAX Networks: A Fold-and-Demultiplex Method,” submitted to *IEEE Trans. on Mobile Computing*.

2. J.-J. Chen, S.-L. Wu, S.-W. Wang, and Y.-C. Tseng, “Per-Flow Sleep Scheduling for Power Management in IEEE 802.16e Networks,” submitted to *IEEE Trans. on Vehicular Technology*.
3. J.-J. Chen, L. Li, and Y.-C. Tseng, “Integrating SIP and IEEE 802.11e to Support Handoff and Multi-grade QoS for VoIP Applications,” submitted to *Computer Networks*.
4. J.-M. Liang, J.-J. Chen, Y.-C. Wang, and Y.-C. Tseng, “A Cross-Layer, Low-Complexity Approach for Downlink Fair Scheduling and Burst Allocation in IEEE 802.16 OFDMA Networks,” to be resubmitted to *IEEE Trans. on Wireless Communications*.
5. J.-J. Chen and Y.-C. Tseng, “An Energy Efficient Sleep Scheduling with QoS Consideration for IEEE 802.16e Wireless Networks,” to be submitted.

Submitted Conference Papers

1. J.-J. Chen, J.-M. Liang, and Y.-C. Tseng, “An Energy Efficient Sleep Scheduling Considering QoS Diversity for IEEE 802.16e Wireless Networks,” submitted to *ICC'10*.
2. J.-M. Liang, J.-J. Chen, H.-C. Wu, and Y.-C. Tseng, “A Simple and Regular Scheduling Algorithm in IEEE 802.16 Grid-based Mesh Networks,” submitted to *VTC2010-Spring*.

Proposed IEEE 802.16m Contributions

1. S.-L. Wu, J.-T. Chang, S.-W. Wang, Y.-C. Tseng, J.-J. Chen, Y.-H. Lee, Y.-G. Jan, M.-H. Chuang, and H.-W. Tseng, “Enhance Location update efficiency using LBS with GIS,” IEEE C802.16m-08/1287r3, Nov. 10, 2008.
2. C.-C. Luo, J.-J. Chen, S.-L. Wu, S.-P. Kuo, Y.-C. Tseng, Y.-H. Lee, Y.-G. Jan, M.-H. Chuang, and H.-W. Tseng, “Comment on C80216m-LBS-08_022r1,” C80216m-LBS-08_039r1.doc, Oct. 30, 2008.
3. S.-L. Wu, J.-J. Chen, C.-C. Luo, S.-P. Kuo, J.-T. Chang, S.-W. Wang, and Y.-C. Tseng, “Text Proposal on the Measurement Methods for LBS,” C80216m-LBS-08_029r1.doc, Oct. 22, 2008.

4. S.-L. Wu, J.-T. Chang, S.-W. Wang, Y.-C. Tseng, J.-J. Chen, Y.-H. Lee, Y.-G. Jan, M.-H. Chuang, and H.-W. Tseng, “Enhance Idle Mode Location update efficiency using LBS with GIS,” C80216m-LBS-08.024.doc, Oct. 22, 2008.
5. C.-C. Luo, J.-J. Chen, S.-P. Kuo, S.-L. Wu, and Y.-C. Tseng, “Text Proposal on the Measurement Methods for LBS,” C80216m-LBS-08.027.doc, Oct. 22, 2008.
6. Y.-H. Lee, Y.-G. Jan, H.-W. Tseng, M.- H. Chuang, S.-T. Sheu, W.-E. Chen, S.-L. Wu, and J.-J. Chen, “Comments on 2nd Draft of Multi-Carrier RG Text Proposal,” C80216m-Multicarrier-08.032.doc, Oct. 28, 2008.
7. Y.-G. Jan, Y.-H. Lee, H.-W. Tseng, M.- H. Chuang, S.-T. Sheu, W.-E. Chen, S.-L. Wu, and J.-J. Chen, “Comments on 2nd Draft of Multi-Carrier RG Text Proposal,” C80216m-Multicarrier-08.019.doc, Oct. 26, 2008.
8. Y.-G. Jan, Y.-H. Lee, H.-W. Tseng, M.- H. Chuang, S.-T. Sheu, W.-E. Chen, S.-L. Wu, and J.-J. Chen, “Comments on 2nd Draft of Handover Support in Multi-Carrier RG Text Proposal,” C80216m-Multicarrier-08.020.doc, Oct. 26, 2008.
9. Y.-G. Jan, Y.-H. Lee, H.-W. Tseng, M.- H. Chuang, S.-T. Sheu, W.-E. Chen, S.-L. Wu, and J.-J. Chen, “Comments on 2nd Draft of Handover Support in Multi-Carrier RG Text Proposal,” C80216m-Multicarrier-08.021.doc, Oct. 26, 2008.

